مقدمة

في النظرية الاحتسابية

الدكتور

عبدالحسين محسن عبدالله

استاذ مساعد

قسم علوم الحاسبات/كلية العلوم/جامعة البصرة

2015

تقديم

تستعمل مفاهيم النظرية الاحتسابية في الكثير من التطبيقات العملية، فهي تستعمل في تصميم الدوائر المنطقية وفي تصميم معالجات النصوص والمترجمات والانظمة التي تعتمد على تمييز الانماط مثل معالجة الصور، وغيرها من المجالات. و تعد مادة النظرية الاحتسابية من المواد الأساسية في علم الحاسوب، فهي مادة متفق على أهمية وجودها في أي منهج دراسي يؤدي إلى منح شهادة جامعية في علم الحاسوب، في معظم جامعات العالم. والهدف من تدريس هذه المادة، انها تقدم الاسس النظرية التي قام عليها علم الحاسوب كما انها تعطي الطالب القدرة على التفكير المنطقي في بناء الخوارزميات لانها تحتاج الى قدرات عقلية في التفكير والاستنتاج والاستنباط، أي أنها تحتاج الى عقل ابداعي. وعلى الطالب ان يكون ملما بموضوع الهياكل المتقطعة discrete structures لأنه يعتبر ممهدا لدراسة هذا الموضوع والفصل الاول يشير الى المفاهيم الرياضية الاساسية التي يتناولها موضوع الهياكل المتقطعة.

ونظرا للنقص في المراجع العربية في هذا الموضوع ، فقد رأيت أنه من الضروري إعداد هذا الكتاب لمساعدة الطالب في دراسة مادة النظرية الاحتسابية. و بعد الاطلاع على عدد من مناهج الجامعات العالمية و على عدد من الكتب وجدت ان يكون هذا الكتاب متضمنا الفصول التالية:

الفصل الاول: يتضمن هذا الفصل مجموعة من المفاهيم الرياضية التي تستند عليها النظرية الاحتسابية. فقد ضم هذا الفصل المجموعات و المخططات واللغات والقواعد.

الفصل الثاني: بدأ هذا الفصل في وضع اللبنة الاولى في موضوع النظرية الاحتسابية فتناول آلات الحالات المنتهية بنوعيها المحدد وغير المحدد واللغات التي تميزها و تكافؤ الآلتين.

الفصل الثالث: تناول هذا الفصل الصيغة الجبرية التي تستعمل في وصف كلمات اللغات المنتظمة المسماة بالتعابير المنتظمة و طرائق تحويلها الى آلات الحالات المنتهية.

الفصل الرابع: يتناول هذا الفصل قواعد حرة السياق وسبل حل مشكلة تمييز اللغات غير المنتظمة التي لم تستطع آلات الحالات المنتهية بنوعيها ان تميزها و دراسة آلة الحالات المنتهية ذات الدفع الى الاسفل.

الفصل الخامس: تناول هذا الفصل آلة اكثر تطور من الآلات السابقة تسمى آلة تورنك لديها ذاكرة كبيرة و قدرة على حل المسائل المختلفة ثم تناول الفصل آلات الحالات المنتهية ذات الاخراج التي تسمى بآلة مور و آلة ميلي.

ولا يسعني هنا الا ان اتقدم بالشكر والتقدير الى الاستاذ المساعد الدكتور صباح سليم عبدالقادر لتقويمه الكتاب علميا والشكر موصول الى الاستاذ المساعد الدكتورة رباب حسين لتقويمها الكتاب لغويا.

والله ولي التوفيق

الاهداء

الى زوجتي **نــــا ديـن**

محتويات الكتاب

الفصل الثالث

الفصل الرابع

الفصل الخامس

المصادر

الفصل الثاني

آلات الحالات المنتهية

1.2 مقدمة

تمثل نظرية التشغيل الذاتي theory of automata احد فروع علم الحاسوب النظرية المثيرة. انشأت جذور هذا الفرع خلال القرن العشرين عندما بدأ علماء الرياضيات، نظريا، تطوير آلات تحاكي بعض الصفات لدى الانسان في اجراء العمليات الحسابية بطريقة موثوق بها. إن كلمة "Automaton" نفسها، ترتبط ارتباطا وثيقا بالكلمة "Automation" الأتمتة، التي تدل على التنفيذ التلقائي لعمليات معينة و توليد نتيجة لهذه العمليات. من خلال نظرية التشغيل الذاتي يستطيع علماء الحاسوب ان يفهموا كيف تستطيع الآلة ان تحسب الدوال و تحل المسائل والاهم من ذلك، ماذا يعني ان دالة تعرف بانها قابلة للحساب Computable او لسؤال يوصف بانه قابل للحسم (او يمكن اتخاذ قرار فيه) decidable.

ان الآلات ذاتية التشغيل هي عبارة عن انموذجات رياضية مجردة لآلات لها القدرة على انجاز عمليات احتسابية على مدخلات تتحرك من خلال سلسلة من الحالات. في كل حالة من حالات الاحتساب وهناك دالة انتقال transition function تقوم بتحديد الحالة المقبلة للآلة اعتمادا على حالة الآلة الحالية. و نتيجة لذلك؛ ما ان تصل الآلة الى حالة قبول accepting state تقوم الآلة بقبول المدخلات و تنهي عملها.

إن من اهداف نظرية التشغيل الذاتي theory of automata تطوير الاساليب التي يستعملها علماء الحاسوب لوصف و تحليل السلوك الديناميكي للانظمة المتقطعة discrete systems، التي فيها تأخذ عينات من الاشارات بشكل دوري signal sampling . ويجري تحديد سلوك هذه الأنظمة من خلال الطريقة التي يجري بناؤها بها من ذاكرة ودوائر ارتباطية combinational. وخصائص هذه الآلات تتضمن:

- الادخال Input : يفترض ان يكون بشكل سلسلة من الرموز تنتمي الى مجموعة منتهية A من اشارة الادخال Input signal. وهذه المجموعة هي {X1, X2, . . ., Xn} إذ أن n يمثل عدد المدخلات.
- الاخراج Output: هو سلسلة من الرموز تنتمي الى مجموعة منتهية Z. وهي المجموعة {Y1, Y2, . . ., Ym} إذ أن m تمثل عدد المخرجات.
- الحالات States: وهي المجموعة المنتهية K، التي تعريفها يعتمد على نوع الآلة. هناك اربعة انواع من آلات التشغيل الذاتي:

1. Finite-state machine
2. Pushdown automat

3. Linear-bounded automata
4. Turing machine

ان التاريخ المثير لكيف اصبحت نظرية التشغيل الذاتي فرع من علم الحاسوب يوضحه مجموعة واسعة من التطبيقات. وإن أول الناس إلذين تأملوا في مفهوم آلة الحالات المنتهية ، هو فريق من علماء علم الاحياء Biologists وعلماء النفس psychologists وعلماء الرياضيات mathematicians والمهندسين engineers وبعض علماء الحاسوب الأوائل الذين كان يجمعهم اهتمام مشترك، وهو نمذجة عملية التفكير لدى الإنسان، سواء في المخ أو في جهاز الكمبيوتر. كان وارن ماكلوخ Warren McCulloch و والتر بيتس Walter Pitts، اثنين من علماء الفسيولوجيا العصبية، إذ كانا أول من قدم وصفا لآلة الحالات المنتهية في عام 1943، في بحثهما الموسوم " A Logical Calculus Immanent in Nervous Activity". قدمت هذه الدراسة مساهمات كبيرة لدراسة نظرية الشبكة العصبية و نظرية آلات الحالات المنتهية و نظرية الحساب وعلم التحكم الآلي cybernetics. في وقت لاحق، قام اثنان من علماء الحاسوب جورج ميلي G.H. Mealy و اويلرمور E.F. Moore، بتعميم النظرية الى آلات ذات قدرات اكبر في بحوث منفصلة، نشرت في عامي 1955 و 1956. جرت تسمية آلات الحالات المنتهية، بآلة ميلي وآلة مور، تقديرا لعملهم.

تعرّف الآلة ذاتية التشغيل M رياضيا بأنها الخماسي $M = (K, \Sigma, \delta, S, F)$ إذ أن:

K : تمثل مجموعة الحالات المكونة للالة وهي مجموعة منتهية.

Σ : تمثل مجموعة الابجدية.

δ : هي دالة الانتقال من حالة الى حالة اخرى و تعرف بالصيغة التالية:

$$\delta : K \times \Sigma \rightarrow K$$

أي أن تكون بالصيغة التالية:

$$\delta (a,qi) = q_j$$

إذ أن $a \in \Sigma$ و $q_i, q_j \in K$

S : تمثل حالة البداية initial state ، إذ أن $S \in K$.

F: تمثل مجموعة حالات النهاية Final States، إذ أن $F \subseteq K$.

تستعمل نظرية الآلات الذاتية التشغيل في علوم الحاسوب في امور كثيرة منها:

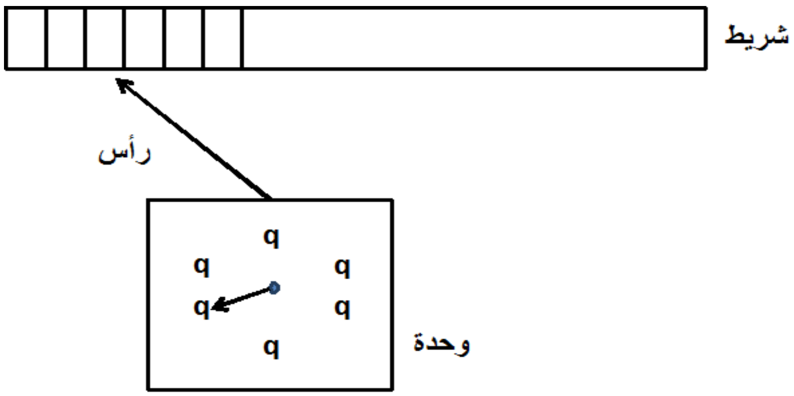
- تطوير برامج للتحقق من صحة وسلامة الدوائر الرقمية.
- محلّل المفردات Lexical Analyzer لمعظم مترجمات لغات البرمجة Compilers؛ أي المكوّن الذي يهتمّ بتحليل النصّ الدّاخل (البرنامج المصدري) Source Program إلى وحدات منطقية (مفردات)؛ كأسماء المتغيرات، والأعداد والتنقيط.
- برامج لتحليل صفحات كثيرة من الإنترنت، والبحث عن كلمات معينة أو انموذجات نصّية.
- برامج للتحقّق من صحّة برامج أخرى، فيما يعرف بالتحقق من الانموذجات (Model Checking).

وسوف نبدأ بأبسط انموذج من انموذجات الآلات ذاتية التشغيل والذي يسمى آلات الحالات المنتهية Finite State Machines.

2.2 آلات الحالات المنتهية Finite State Machines (**FSM**)

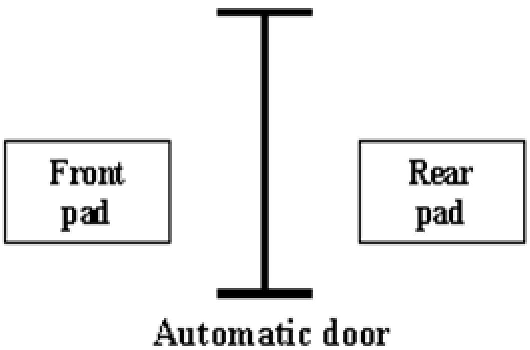
آلة الحالات المنتهية FSM، هي عبارة عن انموذج لجهاز احتسابي بسيط Computational Device. تمتلك هذه الاجهزة حجما صغيرا جدا من الذاكرة و يعالج مدخلاته بصورة مباشرة Online. نعني بهذا أن الجهاز يقرأ رمزا واحدا خلال وحدة الزمن و يقوم بمعالجته. و يمكن النظر الى آلة الحالات المنتهية FSM على انها جهاز يتكون من شريط يوضع فيه الخيط الرمزي string كادخال للجهاز. ويبدأ الجهاز من الحالة الاولى والتي تسمى حالة البداية (q_0)، كما يوجد رأس للقراءة يبدأ القراءة من شريط المدخلات من جهة اليسار متحركا نحو اليمين وفي كل مرة يقرأ رمزا واحدا و يتوقف عند انتهاء الرموز او عند حصول خطأ. وفي كل مرة يقرأ فيها رمزا يدقق في جدول الانتقال Transition Table ليحدد الحالة التي سينتقل اليها استنادا الى الرمز المدخل تبعا لدالة الانتقال $\delta(a,q_i) \rightarrow q_j$ إذ أن a تمثل الرمز المدخل وان q_i تمثل الحالة الحالية و q_j تمثل الحالة التي سينتقل اليها اعتمادا على الرمز المدخل a. الشكل (2-1) يوضح المخطط الذي يمثل آلة الحالات المنتهية المفترضة.

وللتوضيح نأخذ الابواب الآلية Automatic door كمثال عن آلة الحالات المنتهية FSM. والأبواب الآلية نراها عادة في مداخل الفنادق والمطارات والمحلات التجارية وغيرها. يوجد متحكم Controller يقوم بالسيطرة على فتح الباب عند اقتراب جسم مسافة معينة. في هذه الأنواع من الأبواب تكون هناك منطقة أمام الباب في حالة وجود أي جسم



الشكل (2-1) مخطط كتلي لآلة الحالات المنتهية

تسمى (Front Pad) . في هذه المنطقة هناك مجسات تستشعر هذا الجسم وتنقل الإشارة إلى المتحكم Controller الذي يقوم بتحليل وتفسير هذه الإشارة ثم يصدر أمرا إلى الأجزاء الميكانيكية لتقوم بفتح الباب فيكون الباب في حالة Open. كما أن هناك منطقة خلف الباب تسمى (Rear Pad) إذا وجد عندها جسم يقوم المتحكم بإصدار إشارات تبقي الباب مفتوحا بالقدر الكافي من الزمن لكي يعبر هذا الجسم من خلال الباب . وإذا كانت حالة الباب Open ووجود جسم خلف الباب فإن حالة الباب تبقى Open وذلك حتى لا يتعرض هذا الجسم للأذى إذا جرى إغلاق الباب. أي أن مهمة المتحكم هي تغيير حالة الباب من Closed إلى Open والعكس.



الشكل(2-2) مخطط حالة متحكم الباب الالي

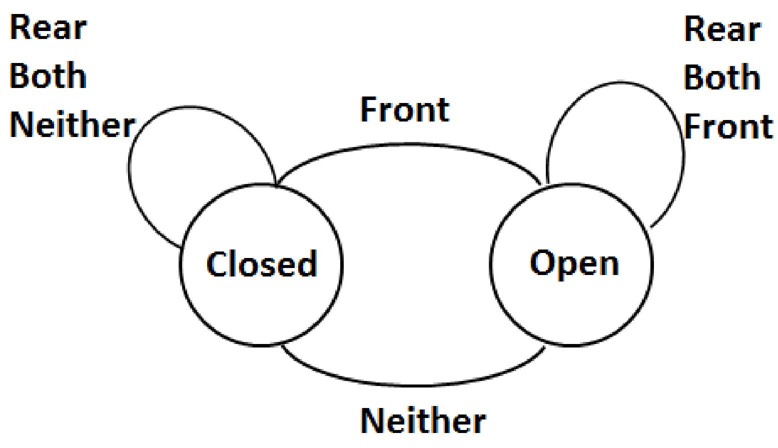
من الشكل (2-2) نلاحظ أن هناك حالتان للمتحكم هي Closed و Open وكذلك أربع

احتمالات للقيم التي يمكن إدخالها إلى هذا النظام .وبناءً على هذه المدخلات يقوم المتحكم بتغيير حالتها من Open إلى Closed وبالعكس .وهذه المدخلات هي:

- امام Front وتعني أن هناك جسما ماديا أمام الباب.
- خلف Rear وتعني أن هناك جسما ماديا خلف الباب.
- الاثنان Both وتعني أن هناك جسمان أحدهما أمام الباب والثاني خلف الباب.
- لا احد Neither وتعني أن لا جسم أمام الباب أو خلفه.

ويوضح المخطط في الشكل (2-3) أنه:

1. إذا كانت الحالة الابتدائية Closed وجرى إدخال إشارة Neither او اشارة Rear فان الباب سيبقى في حالة Closed.
2. و إذا كانت الإشارة Both فلن يفتح الباب وتبقى الحالة Closed حتى لا يتعرض للأذى الشخص الموجود خلف الباب إذا جرى فتحه.
3. في حالة كانت الإشارة المدخلة Front فقط تتغير الحالة إلى Open.
4. أما إذا كانت الحالة الابتدائية Open فإن استقبال الإشارات Both و Front و Rear سيبقي الباب في حالة Open.
5. وفقط في حالة واحدة هي استقبال الإشارةNeither يتم الإغلاق.



الشكل(2-3) مخطط آلة الحالات المنتهية للباب الالي

إن دراسة متحكم الباب الآلي كأنموذج لآلة الحالات المنتهية تعد مفيدة؛ لأنها تقدم طريقة واضحة لتمثيل هذا النوع من النظم كما هو واضح في المثال السابق .فالمتحكم هو عبارة عن حاسوب يمتلك ذاكرة بحجم ثنائية واحدة One Bit يمكنها تسجيل الحالة التي يكون عليها المتحكم هل هي Closed او Open. وهناك بعض الأجهزة التي تمتلك ذاكرة محدودة ولكنها أكبر قليلا من هذه . فمثلا حالة متحكم المصعد الآلي Elevator يمثلها رقم الطابق الذي يوجد عليه المصعد في لحظة ما. أما المدخلات فيمكن أن تكون الإشارات التي يجري إرسالها بالضغط على أزرار موجودة على لوحات للتحكم في العمليات المطلوبة من المصعد القيام بها . وهذا النوع من المتحكمات يوجد في الكثير من الأجهزة المستعملة في البيوت مثل الغسالات وأجهزة التبريد والتكييف وكذلك توجد في الآلات الحاسبة والساعات الإلكترونية وغيرها. ولوصف آلات الحالات المنتهية رياضيا فإنه يجب فعل ذلك بطريقة مجردة أي دون ربط ذلك بأية تطبيقات.

هناك نوعين من آلات الحالات المنتهية FSM هما:

1. **آلات الحالات المنتهية المحددة** Deterministic Finite Automaton(DFA)
2. **آلات الحالات المنتهية غير المحددة** Nondeterministic Finite Automaton(NFA)

3.2 آلات الحالات المنتهية المحددة
Finite Automaton(DFA) Deterministic

تمثل آلات الحالات المنتهية FSM آلة مجردة لها وضع داخلي أو حالة في كل لحظة. تستطيع هذه الآلة قراءة سلسلة من الرموز على التوالي باتجاه واحد (دون إمكانية العودة إلى رمز جرت قراءته سابقاً) والانتقال إلى حالة أخرى حسب الرمز المدخل والحالة الداخلية الحالية للآلة. وتعرف آلات الحالات المنتهية FSM **والتي تسمى آلات الحالات المنتهية المحددة Deterministic Finite Automata** و يرمز لها بالرمز **DFA** ، تعرف رياضيا على انها نظام يتكون من الخماسي:

$$M = (K, \Sigma, \delta, S, F)$$

إذ أن:

K : تمثل مجموعة الحالات المكونة للآلة وهي مجموعة منتهية.

Σ : تمثل مجموعة الابجدية.

δ : هي دالة الانتقال من حالة الى حالة اخرى و تعرف بالصيغة التالية:

$$K \times \Sigma \rightarrow K : \delta$$

أي أن تكون بالصيغة التالية:

$$\delta(a,qi) = qj$$

إذ أن $a \in \Sigma$ و $q_i, q_j \in K$

S : تمثل حالة البداية initial state ، إذ أن $S \in K$.

F: تمثل مجموعة حالات النهاية Final States ، إذ أن $F \subseteq K$.

وان اللغة L المعرفة بآلة الحالات المنتهية M يرمز لها بالرمز L(M) هي اللغة التي تقبل كلماتها (الخيوط الرمزية) من قبل هذه الآلة إذ أنه:

- ✓ **اذا كانت L=L(M) يقال ان الآلة M مميزة Recognizer للغة L.**
- ✓ **اذا ميزت اللغة من قبل آلة الحالات المنتهية المحددة DFA فان اللغة تسمى لغة منتظمة Regular Language. اي لاثبات ان اللغة منتظمة Regular Language يجب بناء DFA بحيث تحقق العلاقة L = L(M).**

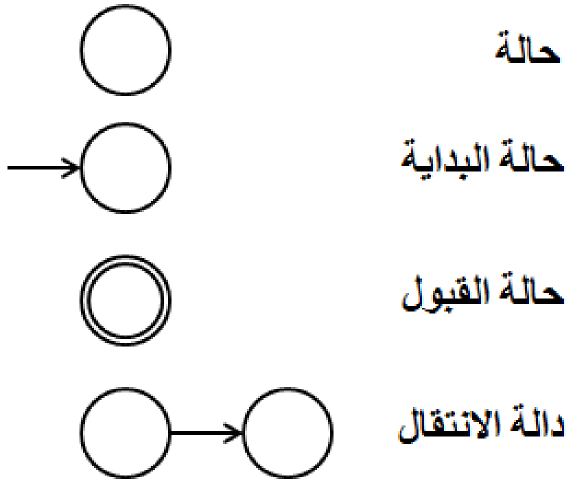
تتصف الآلة DFA بالصفات التالية:

1. لها حالة بداية واحدة.
2. كل حالة تخرج منها مجموعة من الاسهم تمثل انتقالات الآلة من حالة الى اخرى بحيث يكون عددها مساويا الى عدد رموز الابجدية.
3. قد يكون هناك اكثر من حالة قبول واحدة في الآلة.

ملاحظة

اذا قامت الآلتان M1 و M2 بتمييز اللغة L يقال ان الآلة M1 تكافيء الآلة M2.

تستعمل الاشكال الظاهرة في الشكل (2-4) في تمثيل الآلة DFA رسوميا.



الشكل(4-2) الاشكال المستعملة في بناء FSM

إن تصميم الآلة عملية ابداعية تتطلب من المصمم استعمال قدراته العقلية لتفسير المسألة وتحليلها ويمكن للمصمم ان يتصور نفسه يعمل بدلا عن الآلة لكي يستطيع فهم عمل الآلة. فلو اردنا بناء DFA تقوم بقبول لغة معينة فان عليها قراءة الخيوط الرمزية (الكلمات) و تقرر فيما اذا كان الخيط الرمزي الذي قرأته ينتمي الى هذه اللغة ام لا. يجري قراءة الرموز المكونة للخيط الرمزي الواحد تلو الاخر وبعد كل عملية قراءة يجب على الآلة ان تجيب على سؤال: هل ان هذا الخيط الرمزي ينتمي الى هذه اللغة؟. والسبب في ذلك أن الآلة لا تعلم متى تنتهي عملية قراءة الرموز المكونة للخيط الرمزي لذا يجب عليها ان تكون جاهزة للاجابة على هذا السؤال وفي اية لحظة. ولكي تستطيع الآلة ان تجيب على السؤال يجب عليها معرفة ماهي الاشياء التي يجب ان تحتفظ بها. و بما ان هذا النوع من الآلات ذاكرته صغيرة جدا لانها تمتلك عددا محدودا من الحالات States، فانها لا تستطيع الاحتفاظ بكمية كبيرة من البيانات.

لبناء الة DFA يجب:

اولا: تحليل المسألة المعطاة و تحديد الصفات البارزة في المسألة و جعل كل صفة تمثل بحالة State .

ثانيا: تحديد العوامل التي تؤدي الى الانتقال من حالة الى اخرى ضمن الآلة، اي ضبط عملية الانتقال اعتمادا على المدخلات.

ثالثا: تحديد حالة النهاية (حالة قبول المدخل) Accepting State او حالات النهاية اذا كان هناك اكثر من احتمال واحد لقبول المدخل.

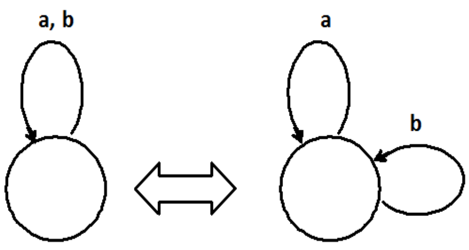
رابعا: تحديد حالة البداية (Start Sate).

خامسا: رسم المخطط الذي يمثل الآلة باستعمال الاشكال الموضحة بالشكل (4-2)

سادسا: اختبر الآلة عن طريق فحص الخيوط الرمزية (الكلمات) التي تنتمي الى اللغة المراد تمييز ها.

ملاحظات

- ✓ في آلة الحالات المنتهية المحدودة، من اجل كل حالة ورمز ادخال توجد قيمة وحيدة لدالة الانتقال.
- ✓ يمكن تمثيل آلة الحالات المنتهية المحدودة بمخطط موجه له اسهم تمثل الانتقالات وعقد التي هي حالات الآلة.
- ✓ عندما يوجد انتقال من حالة الى اخرى فاننا نمثل ذلك بسهم موجه مرفق برمز الادخال الذي يسبب الانتقال.
- ✓ نميزالحالة الابتدائية بوضع سهم قبلها و الحالة النهائية بوضع دائرتين متداخلتين.
- ✓ اذا وجدت لدينا حالة لاتؤدي الى الحالة النهائية فهي تدعى الحالة الرفض Reject State.
- ✓ ليس من الضروري ان تكون الحالة النهائية هي نفسها الحالة البداية.
- ✓ نقول عن خيط رمزي ما انه مقبول من قبل آلة الحالات المنتهية المحدودة، M، اذا وصلت الآلة الى حالة نهائية (حالة قبول) بعد قراءة الخيط باكمله.
- ✓ كل من الشكلين ادناه متكافئان.



مثال1

صمم آلة الحالات المنتهية المحددة DFA لتمييز اللغة التي كلماتها عبارة عن خيوط رمزية ثنائية عدد الآحاد فيها فردي Odd والمعرفة بالصيغة التالية:

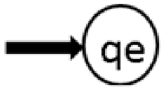
L = {w : w is a binary string containing an **odd** number of 1s**}**

الحل

لاثبات ذلك يجب بناء آلة DFA ،تسمى M، بحيث تحقق L=L(M).

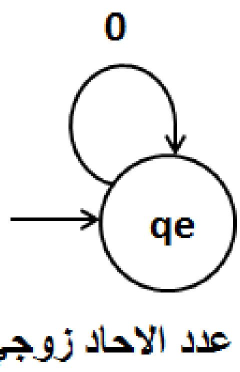
الفكرة: إن الآلة DFA تقرأ الكلمة المدخلة w من اليسار إلى اليمين، وتتبع عدد الآحاد التي تحتويها الكلمة w. بعد قراءة الكلمة w بأكملها، تقوم الآلة DFA بالتحقق فيما إذا كان عدد الآحاد فرديا ام لا، فاذا كان عدد الآحاد فرديا يجري قبول الكلمة w و بعكسه ترفض.

لتصميم هذه الآلة نرى أن الصفة الوحيدة للكلمات (الخيوط الرمزية) في اللغة المراد تمييزها (او التعرف عليها) هي أن عدد الآحاد فيها هو عدد فردي، أي أن المعلومة التي يجب على الآلة الاحتفاظ بها او تذكرها هي هذه الصفة. في حالة البداية Start State نعرف أن الآلة لم تقرأ اي رمز من الرموز المدخلة حتى تلك اللحظة و بذلك يكون عدد الآحاد عند هذه الحالة يساوي صفرا و عليه يكون عددا زوجيا. و هنا نمثل الحالة بالشكل التالي و نطلق عليها اسم **qe** للدلالة على صفة ان عدد الآحاد هو عدد زوجي.

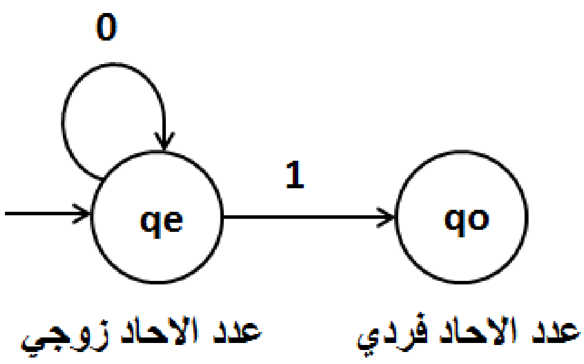


زوجي # 1's

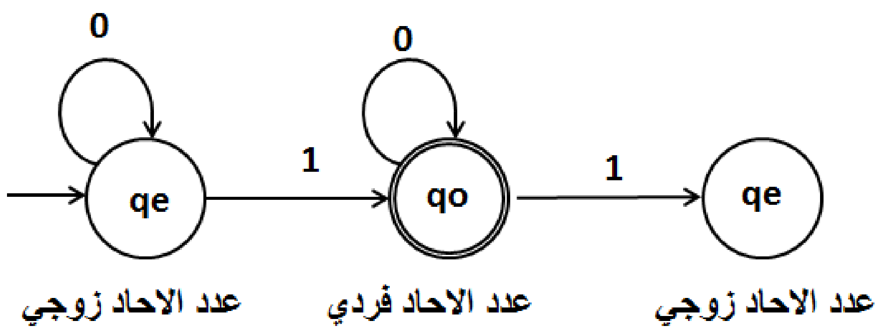
عندما تبدأ الآلة بالقراءة ويكون الرمز الداخل الى الآلة يساوي (**0**) فان هذا الرمز المدخل سوف لا يؤثر على قيمة عدد الآحاد ومن ثم لا يدعو لتغيير حالة الآلة بل تبقى على حالتها الحالية، اي الحالة **qe** كما في الشكل التالي ويشير السهم المنحني لبقاء الآلة في الحالة نفسها التي كان عليها دون انتقال:



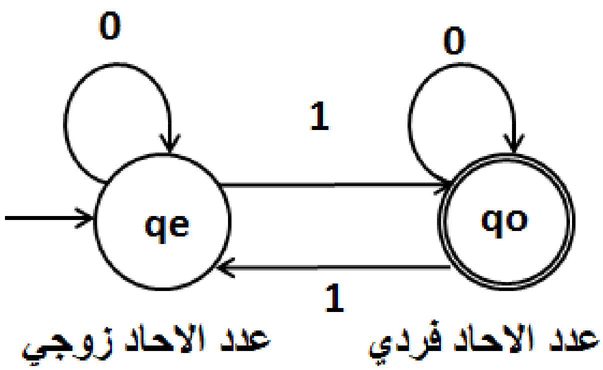
اما اذا كان الرمز المدخل يساوي (**1**) فانه سوف يجعل عدد الآحاد يتغير من زوجي الى فردي. و بذلك سوف نحتاج الى حالة جديدة لتمثيل الاحتمال الجديد الذي تعرضت له الآلة و سوف نشير الى الحالة الجديدة بالاسم **qo** للدلالة على ان عدد الآحاد فردي والشكل التالي يوضح ذلك.



هنا نتسائل من جديد ماذا يحصل للالة لو قرأت رمزا جديدا ان كان صفرا او واحدا وهي في حالتها الجديدة التي انتقلت اليها، اي الحالة qo. طبعا كما حصل في الحالة الاولى : اذا كان الرمز المدخل يساوي (**0**) فان الآلة تبقى في حالتها الحالية و لا تجري أي أنتقال لان الرمز المدخل لا يؤثر على قيمة عدد الآحاد، اما اذا كان الرمز المدخل يساوي (**1**) فان عدد الآحاد هنا سوف يتغير من فردي الى زوجي ومن ثم سوف تنتقل الآلة الى حالة جديدة هي حالة عدد الآحاد يكون زوجيا كما يوضحه الشكل التالي:



نجد هنا أن الحالة الجديدة هي نفسها الحالة الاولى ومن ثم لا ضرورة لاضافتها (وهنا يمكن اعادة استعمال الحالات الموجودة كلما دعت الضرورة الى ذلك) وانما نعيد رسم الشكل اعلاه ليكون بالصيغة الاتية:



الشكل اعلاه يمثل تعريف آلة الحالات المنتهية المحددة **M = (K, Σ, δ, S, F)** بما يسمى مخطط الحالات State Diagram. إذ أن السهم الداخل الى الحالة qe من جهة اليسار يوضح أن هذه الحالة هي حالة البداية Start State ولما كانت الحالة التي يصبح فيها عدد الآحادعددا فرديا هي الحالة التي تقبل فيها الكلمة المدخلة الى الآلة، فان هذه الحالة تسمى حالة القبول Accepting State اوحالة النهاية وتمثل بدائرتين متداخلتين. اما دالة الانتقال **δ** من حالة الى اخرى فتعرف اعتمادا على جدول الانتقالات التالي:

	0	1
qe	**qe**	**Qo**
qo	**qo**	**qe**

و بذلك فان الآلة تعرف شكليا بالصيغة التالي:

M = (K, Σ, δ, s, F)

إذ أن:

1. K={qe, qo}
2. Σ = {0, 1}
3. S = qe
4. F = qo
5. δ:0xqe → qe, δ:0xqo → qo, δ:1xqe → qo,   δ:1xqo→ qe .

في المثال التالي تستعمل الآلة لاكتشاف تسلسل معين من الرموز في كلمات اللغة.

مثال2

صمم آلة الحالات المنتهية المحددة DFA لتمييز اللغة التي كلماتها عبارة عن خيوط رمزية ثنائية وتحتوي على الخيط الرمزي الجزئي 101 والمعرفة بالصيغة التالية:

L = {w : w is a binary string containing **101** as a substring**}**

الحل

لتصميم الآلة DFA المسماة M، لتمييز اللغة المعرفة اعلاه نجد مايلي:

إن الصفة المهمة في هذه اللغة أن أي كلمة فيها يجب أن تحتوي على الخيط الرمزي الجزئي substring (101) وبذلك فان الآلة تبحث عن هذا الخيط الرمزي الجزئي. فان وجدت الآلة الخيط الرمزي الجزئي (101) فانها سوف تقبل الكلمة و بعكسه سوف ترفضها. لذا:

- عندما تبدأ الآلة بقراءة الكلمة المدخلة (الخيط الرمزي) w من اليسار الى اليمين فانها سوف لا تقوم بأي أنتقال من حالة البداية Start State عندما يكون الرمز المدخل قيمته (0) لان الصفة المميزة هنا تبدأ بالرمز 1.
- اذا كان الرمز المدخل قيمته واحد فان الآلة ستضطر الى الانتقال الى حالة جديدة لعل ما سيأتي بعده هو الرمزان (01) التي تشكل الصفة المميزة.
  - اذا كان الرمز التالي يساوي (1) فان الآلة تبقى في الحالة الحالية لان الآلة تريد الرمز 0 حتى تغير حالتها اعتمادا على الصفة المميزة منتظرة لعل ما سيأتي هو الرمزان (01).
  - اما اذا كان الرمز المدخل يساوي (0) فان الآلة سوف تنتقل الى حالة جديدة لعل الرمز التالي يساوي (1) و بذلك تحصل الآلة على الصفة المميزة.
    - اذا كان الرمز التالي يساوي (1) فان الآلة سوف تقبل الكلمة ويجري الانتقال الى حالة جديدة تسمى حالة قبول Accepting State (مع الاستمرار بقراءة بقية رموز الكلمة حتى نهايتها).
    - اما اذا كان الرمز التالي يساوي (0) فتقوم الآلة بالانتقال الى حالة البداية لتجري الخطوات السابقة نفسها.

و تعرف الحالات الاربع التي ذكرناها في الشرح الاعلاه بالشكل التالي:

q1: تكون الآلة M في هذه الحالة عندما يكون آخر رمز جرت قراءته هو الرمز (1) لكن الخيط الرمزي الجزئي (101) لم يقرأ لحد الان.

q10: تكون الآلة M في هذه الحالة عندما يكون اخر رمزين جرت قرائتهما هما (10) لكن الخيط الرمزي الجزئي (101) لم يقرأ لحد الان.

q101: تكون الآلة M في هذه الحالة عندما الخيط الرمزي الجزئي (101) قد جرت قراءته.

q: تكون الآلة M في هذه الحالة في جميع الاوضاع ماعدا الاوضاع السابقة المذكورة في الحالات الثلاث اعلاه.

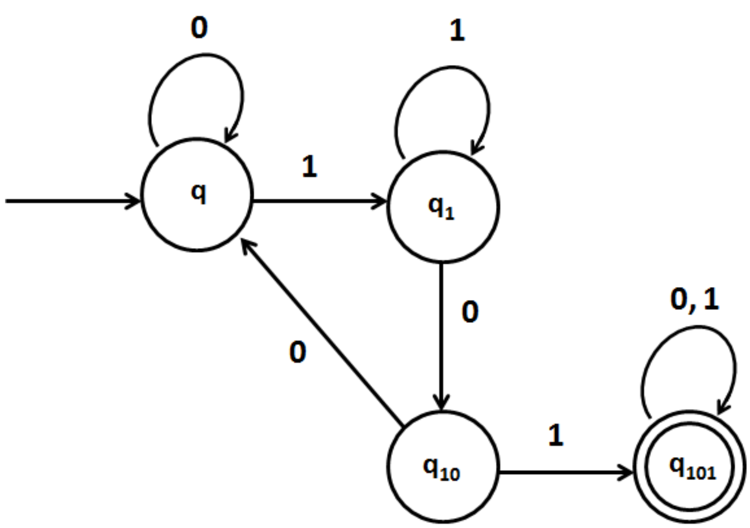
وتعرف هذه الآلة شكليا كما يلي:

M = (K, Σ, δ, s, F)

1. $K = \{q, q_1, q_{10}, q_{101}\}$,
2. $\Sigma = \{0, 1\}$,
3. the start state is q,
4. the set F of accept states is equal to $F = \{q_{101}\}$, and
5. the transition function δ is given by the following table:

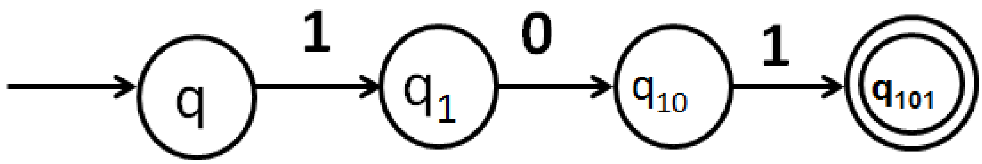
	0	1
Q	q	q_1
q_1	q_{10}	q_1
q_{10}	q	q_{101}
q_{101}	q_{101}	q_{101}

و تمثل بمخطط الحالات التالي:



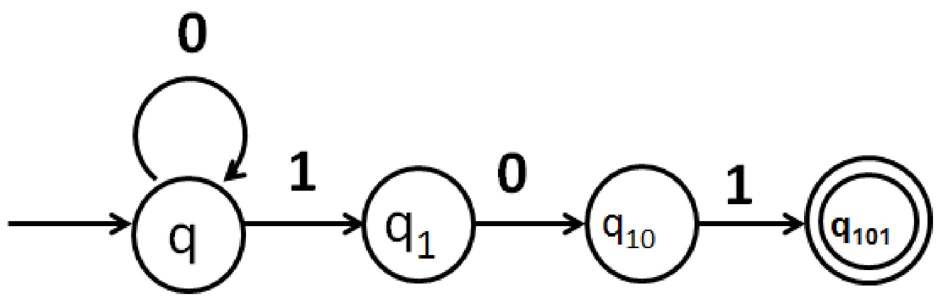
و يمكن تصميم الآلة بالطريقة التالية:

نتساءل ماهي الصفة المميزة للغة، وفي اللغة قيد الدراسة تكون الصفة المميزة هي احتواؤها على الخيط الرمزي الجزئي (101) وعندها نختار مجموعة من الحالات لتمثيل الخيط الرمزي الجزئي (101) تبدأ بحالة البداية و تنتهي بحالة القبول. في حالتنا هذه نختار 4 حالات كما في الشكل ادناه.

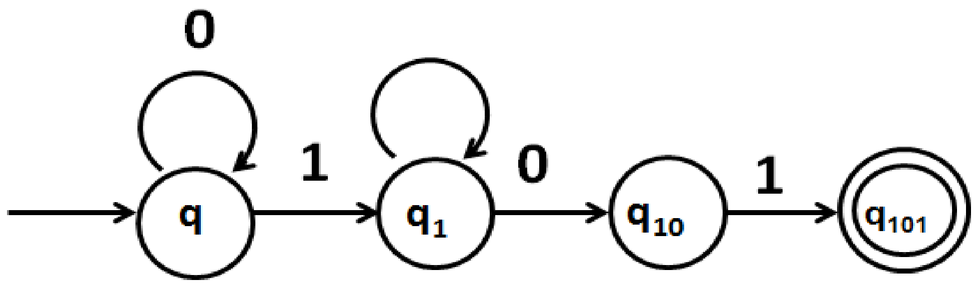


عند تصميم DFA يجب أن يخرج من كل حالة عدد من الاسهم يساوي عدد رموز ابجدية اللغة لتمثل الانتقالات الممكنة من الحالة الحالية الى الحالة الجديدة اعتمادا على الرمز المدخل. وبما ان ابجدية اللغة تتكون من رمزين فقط هما (0) و (1) لذا يجب ان نكمل المخطط بحيث نجعل كل حالة لها انتقالتان واحدة اعتمادا على الرمز (0) والثانية اعتمادا على الرمز (1).

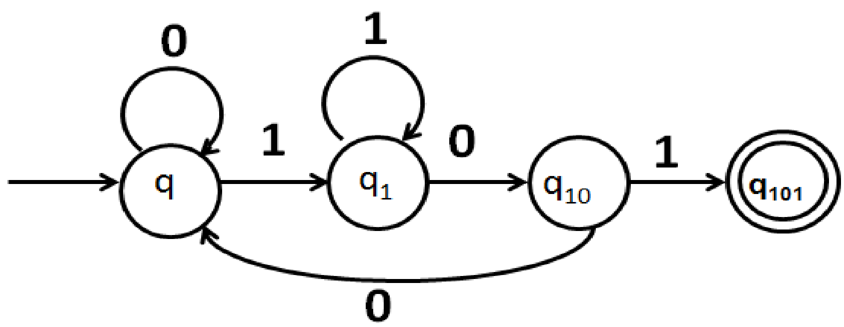
من الشكل اعلاه، الحالة الاولى هي حالة البداية q : نجد انه يخرج منها انتقالة واحدة اعتمادا على الرمز (1) لذا ينقصها انتقالة اخرى اعتمادا على الرمز (0) وبما أن قراءة الرمز (0) لا تؤثر على الشرط المطلوب وهو ظهور الرمز (1) لذا لا يوجد انتقالة الى حالة جديدة و تبقى الآلة في الحالة الحالية كلما ظهر الرمز (0). فتكون الانتقالة من الحالة الحالية (حالة البداية) الى الحالة نفسها و تؤشر في الرسم بالسهم المنحني.



الحالة الثانية الحالة q_1: نجد انه يخرج منها انتقالة واحدة اعتمادا على الرمز (0) لذا ينقصها انتقالة اخرى اعتمادا على الرمز (1) وبما ان قراءة الرمز (1) لا تؤثر لذا لا يوجد انتقالة الى حالة جديدة و تبقى الآلة في الحالة الحالية كلما ظهر الرمز (1) . فتكون الانتقالة من الحالة الحالية (q_1) الى نفسها و تؤشر في الرسم بالسهم المنحني.

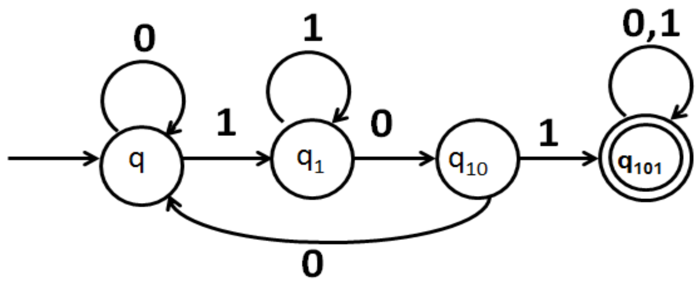


الحالة الثالثة q_{10}: في هذه الحالة يخرج منها انتقالة واحدة اعتمادا على الرمز (1) لذا ينقصها انتقالة اخرى اعتمادا على الرمز (0) وبما ان قراءة الرمز (0) سوف تؤثر على التسلسل الذي وجدته الآلة من الحالات السابقة لغرض الحصول على الصفة (101)، لذلك على الآلة الانتقال الى حالة تسمح لها البحث من جديد عن هذه الصفة. لذا فان الانتقال الى الحالة نفسها او الحالة q_1 لا تسمح للالة بان تحصل على الصفة المطلوبة لذا يجري الانتقال الى حالة البداية لكي تحصل على الصفة (101) بطريقة صحيحة من جديد. كما موضح بالشكل التالي.



الحالة الاخيرة الحالة q_{101}: عند هذه الحالة تكون الآلة قد حصلت على الصفة المطلوبة و هي الحصول على الخيط الرمزي الجزئي (101) لكن لا تحصل منها أي أنتقالة الى حالة جديدة.

لذا على الآلة هنا استكمال قراءة بقية رموز الخيط الرمزي المدخل. و لاتمام هذه العملية فانه لا ضرورة لانتقال الآلة من هذه الحالة الى حالة جديدة و بذلك فالآلة تقرأ بقية الرموز مع بقائها في الحالة نفسها التي هي فيها. والسهم المنحني يمثل الانتقالات ان كان الرمز المدخل (1) او (0).



مثال3

صمم الة حالات منتهية محددة DFA لتمييز اللغة المعرفة ادناه:

L = { w | w is a multiple of 3 when interpreted as a binary integer**}**

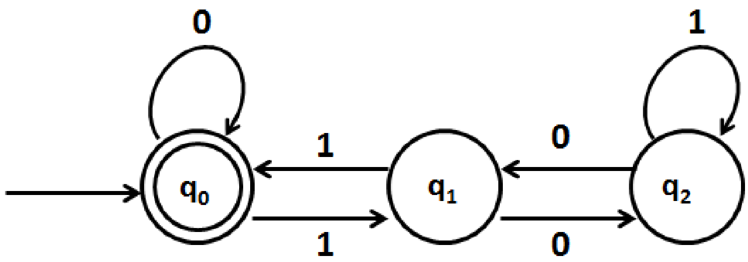
الحل

المطلوب هنا بناء آلة تميز اللغة التي كلماتها هي الأعداد التي تقبل القسمة على 3 او يمكن التعبير عنها بانها الكلمات التي من مضاعفات العدد 3.
لبناء مثل هذه الآلة يطرح السؤال التالي: ماهي الصفات التي تميز كلمات هذه اللغة؟ ان قلنا ان كلمات هذه اللغة تقبل القسمة على 3 بدون باق ، فان عملية القسمة لها ناتج ولها باق ان الناتج لا يمكن ان يساعدنا على التمييز لان هناك مالانهاية من القيم تظهر كناتج لتقسيم الاعداد على 3. اما اذا نظرنا الى باقي القسمة فنجد ان اي عدد يقسم على 3 فان الباقي اما 0 او 1 او 2 ومن ثم يمكن اخذ هذه الظاهرة كصفة تساعدنا على بناء الآلة. اذن يمكن ان نحدد 3 حالات بالشكل التالي:

- الحالة الاولى التي تمثل باقي القسمة يساوي 0 و نرمز لها ب q_0 . وفي هذه الحالة تنتهي عندها كل الأعداد التي هي من مضاعفات العدد 3 وهي الاعداد التي تمثل الكلمات التي تنتمي الى اللغة قيد الدراسة، أي أنها تمثل حالة قبول.
- الحالة الثانية التي تمثل باقي القسمة يساوي 1 و نرمز لها ب q_1 . وفي هذه الحالة تنتهي عندها كل الاعداد التي باقي قسمتها على 3 يساوي 1. مثل العدد 1، 4، 7،. . .

- الحالة الثالثة التي تمثل باقي القسمة يساوي 2 و نرمز لها ب q_2 . وفي هذه الحالة تنتهي عندها كل الاعداد التي باقي قسمتها على 3 يساوي 2. مثل الاعداد 2، 5، 8، 11، . . .

و بذلك يكون مخطط هذه الآلة كما في الشكل التالي:



في هذا الشكل الحالة q_0 تمثل حالة بداية و حالة قبول في الوقت نفسه.

مثال4

صمم الة حالات منتهية محددة DFA تقوم بتمييز اللغة التي كلماتها تحتوي على عدد زوجي من الرمز a و عدد زوجي من الرمز b، كما معرفة بالصيغة ادناه:

L = { w | w has even # of a’s and even # 0f b’s**}**

الحل

كما قلنا في كل مرة اننا عندما نفكر بتصميم الة حالات منتهية محدودة DFA لابد من ان نبحث عن الصفة المميزة للخيط الرمزي التابع للغة المراد تمييز كلماتها. من خلال السؤال، أن الصفة المميزة لكلمات اللغة هي ان عدد الرمز a في الكلمة هو عدد زوجي وكذلك الحال بالنسبة للرمز b. ان اي كلمة في هذه اللغة تكون في واحدة من اربعة احتمالات بالنسبة لعدد الرمزين a و b وهي.

1. **عدد الرمز a يكون زوجيا Even وعدد الرمز b يكون زوجيا Even.**
2. **عدد الرمز a يكون زوجيا Even وعدد الرمز b يكون فرديا Odd.**
3. **عدد الرمز a يكون فرديا Odd وعدد الرمز b يكون زوجيا Even.**
4. **عدد الرمز a يكون فرديا Odd وعدد الرمز b يكون فرديا Odd .**

بمعنى ان الآلة سوف تكون في واحدة من اربع حالات اعتمادا على الاحتمالات الاربعة اعلاه. وتكون هذه الحالات كما يلي:

1. **EE** تكون فيها الآلة عندما عدد الرمز a يكون زوجيا Even وعدد الرمز b يكون زوجيا Even.
2. **EO** تكون فيها الآلة عندما عدد الرمز a يكون زوجيا Even وعدد الرمز b يكون فرديا Odd.
3. **OE** تكون فيها الآلة عندما عدد الرمز a يكون فرديا Odd وعدد الرمز b يكون زوجيا Even.
4. **OO** تكون فيها الآلة عندما عدد الرمز a يكون فرديا Odd وعدد الرمز b يكون فرديا Odd.

عندما تبدأ الآلة عملها فانها لا تقرأ اي رمز في حالة البداية ومن ثم فان عدد الرمز a يساوي صفرا وكذلك هو الحال بالنسبة للرمز b. ولما كان الصفر عددا زوجيا لذا فان عدد الرمز a يكون زوجيا وكذلك هو الحال مع عدد الرمز b، أي أن الآلة تكون في الحالة EE. وبذلك تكون حالة البداية Start State هي الحالة EE. يخرج من الحالة EE سهمان يمثلان الانتقالات اعتمادا على رمز المدخلات الذي تقرأه الآلة. السهم الاول يمثل الانتقالة عندما تقرأ الآلة الرمز a، بينما السهم الثاني يمثل الانتقالة عندما يكون الرمز الذي تقرأه الآلة هو الرمز b.

الان، اذا كان الرمز الذي تقرأه الآلة و هي في الحالة EE هو الرمز a فهذا يعني ان عدد الرمز a اصبح يساوي 1 وهو عدد فردي بينما لم يتغير عدد الرمز b، اي مازال زوجيا. عندها فان الآلة تنتقل من الحالة EE الى الحالة OE. اما اذا كان الرمز الذي تقرأه الآلة هو الرمز b فهذا يعني ان عدد الرمز b اصبح يساوي 1 وهو عدد فردي بينما لم يتغير عدد الرمز a، اي مازال زوجيا. عندها فان الآلة تنتقل من الحالة EE الى الحالة EO.

اذا افترضنا الان ان الآلة اصبحت في الحالة OE، أي أن عدد الرمز a فرديا وعدد الرمز b زوجيا. فاذا قرأ الآلة الرمز a فان عدد هذا الرمز سوف يزداد بمقدار 1، اي يتحول من فردي الى زوجي بينما عدد الرمز b يبقى زوجيا. ان الانتقالة هنا سوف تكون من الحالة OE الى الحالة EE. اما اذا كان الرمز المقروء هو الرمز b فهذا يعني أن عدد هذا الرمز سوف يزداد بمقدار 1، اي يتحول من زوجي الى فردي بينما عدد الرمز a يبقى فرديا. إن الانتقالة هنا سوف تكون من الحالة OE الى الحالة OO.

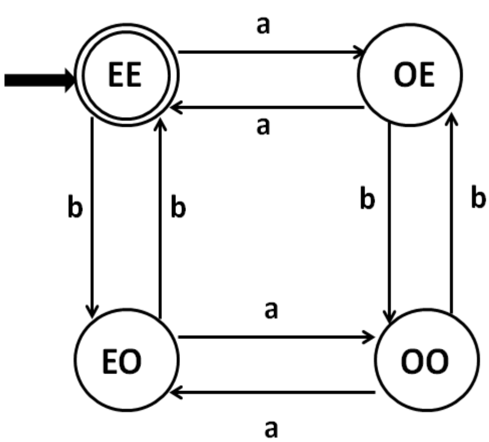
اذا كانت الآلة في الحالة EO، أي أن عدد الرمز a زوجيا وعدد الرمز b فرديا. عندما تقرأ الآلة الرمز a فان عدده سيزداد بمقدار 1، اي يتحول من زوجي الى فردي اما عدد الرمز b فيبقى على حاله، اي فرديا وبذلك تنتقل الآلة من الحالة EO الى الحالة OO. اما اذا قرأت

الآلة الرمز b فان عدد هذا الرمز سيزداد بمقدار 1 ويصبح زوجيا بينما يبقى عدد الرمز a زوجيا، أي أن الآلة تنتقل من الحالة EO الى الحالة EE.

اما الحالة الاخيرة وهي الحالة OO، أي أن عدد الرمز a عدد فردي وكذلك هو حال الرمز b، فان الآلة عندما تقرأ الرمز a فان عدده سيصبح زوجيا بينما لا يتغير عدد الرمز b وبذلك فان الآلة تنتقل من الحالة OO الى الحالة EO. اما اذا كان الرمز المقروء من قبل الآلة هو الرمز b فان عدده سيصبح زوجيا و يبقى عدد الرمز a على حاله فتكون الانتقالة عندها من الحالة OO الى الحالة OE. و جدول الانتقالات التالي يوضح دالة الانتقال في الآلة.

	a	b
EE	OE	EO
EO	OO	EE
OE	EE	OO
OO	EO	OE

و مخطط الحالات لهذه الآلة كما في الشكل التالي:



ان حالة القبول هنا هي الحالة EE.

- **جرب تصميم الة DFA عندما تكون كلمات اللغة في واحدة من الاحتمالات الثلاثة الاخرى.**

مثال5

لتكن {0, 1} هي الابجدية المعرف عليها اللغة L بحيث جميع الخيوط الرمزية التي تنتمي اليها تحقق الشروط الثلاثة التالية:

a. لا تبدأ بصفر.
b. اذا عد الخيط الرمزي w عددا ثنائيا، فانه عدد زوجي.
c. العدد w لا يقبل القسمة على 4.

صمم الآلة DFA التي تقبل الخيوط الرمزية التي تنتمي الى L. مثلا العدد 10010 ينتمي الى اللغة بينما الاعداد 101 و 000010 و 10000 لا تنتمي الى اللغة.

الحل

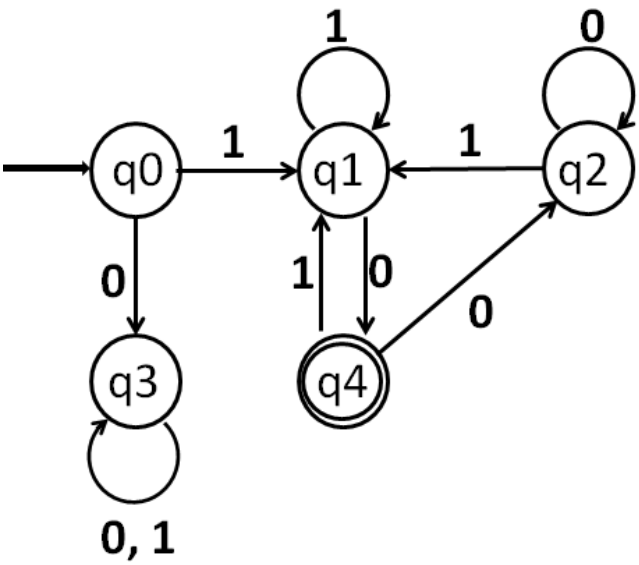
إن الخيط الرمزي (العدد الثنائي) الذي يحقق الشروط الثلاثة يجب أن يبدأ بالرمز 1 و ينتهي بالرموز 10. اذا لم يبدأ بالرمز 1 فانه لا يحقق الشرط الاول. الشرط الثاني ينص على ان العدد يجب ان ينتهي بالرمز 0 حتى يكون زوجيا حسب التمثيل الثنائي للعدد. اما الشرط الثالث فانه ينص على ان الخيط الرمزي يجب ان ينتهي بالرموز 00 لانه ان احتواها فان العدد سوف يكون قابلا للقسمة على 4 حسب تمثيل الاعداد في النظام الثنائي. لذا من الشرط الثاني والثالث نستنتج ان الخيط الرمزي يجب ان ينتهي بالرموز 10.

لتصميم الآلة DFA نختار حالة بداية مثل الحالة q0. عند قراءة الرمز الاول وكان هو الرمز 0 فان الآلة ترفض الخيط الرمزي لانه لا يبدأ بالرمز 1، لذا تنتقل الآلة من الحالة q0 الى الحالة q3 التي تمثل حالة الرفض Reject في الآلة (لم نستعمل مثل هذه الحالة في الامثلة السابقة فقط حتى لا يكبر حجم الآلة، لكن هنا استعملت لغرض توضيح انه يمكن ان تمتلك الآلة حالة رفض مثلما تمتلك حالة قبول). وعند الحالة q3 تقرأ بقية الخيط الرمزي حتى تتوقف. اما اذا كان الرمز المقروء هو الرمز 1 فان الآلة تنتقل من الحالة q0 الى الحالة q1. و بذلك يتحقق اول شرط.

عند الحالة q1، اذا كان الرمز المقروء هو الرمز 1 فلا ضرورة لانتقال الآلة الى حالة جديدة لانها لا تؤثر على عمل الآلة. اما اذا كان الرمز 0 فان الآلة تنتقل الى الحالة q4 فاذا كان الخيط الرمزي طوله يساوي 2، اي يمثل الخيط الرمزي 10 فان الآلة سوف تقبله لانه يحقق الشروط الثلاثة وبذلك تكون الحالة q4 هي حالة القبول للالة. اما اذا كان طول الخيط الرمزي اكثر من 2 فعلى الآلة ان تخرج من الحالة q4 الى حالة اخرى، وافضل حالة يمكن ان تنتقل اليها هي الحالة q1 عندما يكون الرمز المقروء هو الرمز 1 حتى تحافظ الآلة على الشرط

الثاني والثالث (اي ينتهي الخيط الرمزي ب 10). اما اذا كان الرمز المقروء عند الحالة q4 هو الرمز 0 فان الآلة سوف تنتقل الى حالة جديدة مثل q2.

عندما تكون الآلة في الحالة q2 فانها ستنتقل الى الحالة q1 عندما يكون الرمز المقروء هو 1 حتى تساعد على امكانية الحصول على الشرط الثاني والثالث، اما اذا كان الرمز المقروء هو الرمز 0 فانه لا يؤثر على عمل الآلة ومن ثم لا تحتاج الآلة الى الانتقال الى حالة جديدة. والشكل التالي يمثل الآلة التي تقبل الخيوط الرمزية لهذه اللغة.



حاول حل المسائل التالية :

1. صمم الآلة DFA لتمييز اللغة التي كلماتها تحتوي على الخيط الرمزي الجزئي 0011.
2. صمم الآلة DFA لتمييز اللغة التي كلماتها تحتوي على الخيط الرمزي الجزئي 01.

مثال6

تصميم آلة بيع بسيطة Simple Vending Machine

افرض ان المطلوب هو تصميم آلة بيع بسيطة لعلب العصير. سعر العلبة دولارا واحدا. والآلة تقبل عملة من فئة ربع دولار (25 سنت) و نصف دولار (50 سنت). والآلة تسمح بخروج

العلبة عندما يكون المبلغ يساوي دولارا واحدا او اكبر منه و لا تعيد المتبقي من المبلغ.

الحل

لتصميم هذه الآلة يجب اولا تحديد الحالات التي يمكن ان تكون عليها الآلة عند عملها اعتمادا على المدخلات وكذلك على الحالة الحالية للآلة.

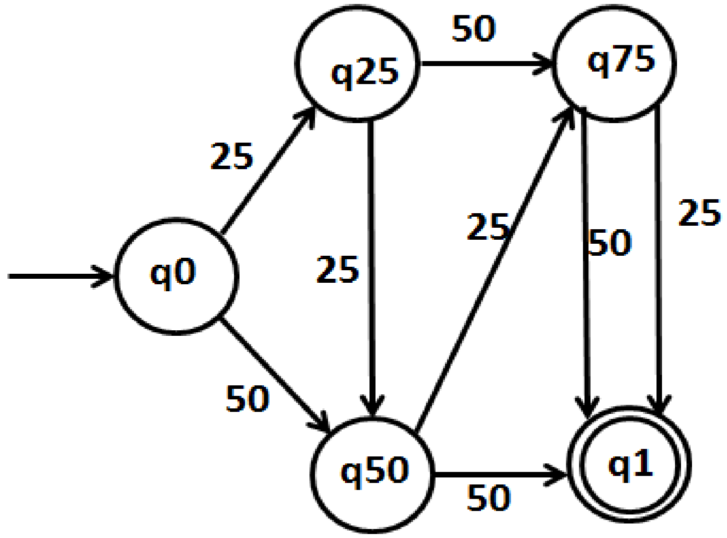
عند بداية عمل الآلة فانها لا تستلم اي مبلغ وبذلك تكون المدخلات تساوي صفرا، أي أن الحالة الاولى التي تمر بها الآلة عند بداية عملها هي الحالة التي تكون المدخلات عندها تساوي صفرا، نسميها q0 وهي تمثل حالة البداية.
عندما يقوم الزبون بادخال عملة من فئة ربع دولار وهي في حالة البداية q0 فان الآلة سوف تنتقل الى حالة جديدة يكون المبلغ الذي استلمته الآلة يساوي ربع دولار، سنسميها q25. اما اذا ادخل الزبون مبلغ نصف دولار و الآلة في حالة البداية فان الآلة سوف تنتقل الى حالة جديدة يكون فيها المبلغ الذي استلمته يساوي نصف دولار وهذه الحالة سوف نسميها q50.

الان عندما تكون الآلة متوقفة عند الحالة q25 وادخل الزبون عملة من فئة ربع دولار فان المبلغ الذي استلمته الآلة يصبح نصف دولار وعندها سوف تنتقل الآلة الى الحالة التي يكون المبلغ فيها يساوي نصف دولار وهي الحالة q50. اما اذا كان المبلغ المدخل هو نصف دولار فان الآلة ستنتقل الى حالة جديدة يكون فيها المبلغ مساويا 75 سنتا، وسنسمي هذه الحالة q75.

اذا كانت الآلة عند الحالة q50 وكان المبلغ المدخل من قبل الزبون يساوي ربع دولار فان مجموع المبلغ المستلم من قبل الآلة هو 75 سنتا وبذلك تنتقل الآلة الى الحالة q75 اما اذا كان المبلغ المدخل يساوي نصف دولار فان الآلة ستنتقل الى حالة جديدة يكون فيها المبلغ المستلم يساوي دولارا واحدا وهي الآلة q1.

عندما تكون الآلة متوقفة عند الحالة q75 وادخل الزبون مبلغ ربع دولار او نصف دولار فان المبلغ المستلم من قبل الآلة يصبح مساويا الى دولار واحد او اكثر وفي كلتا الحالتين سوف تنتقل الآلة الى الحالة q1.

ان المطلوب من الآلة عندما تستلم مبلغ دولار واحد او اكثر ان تقوم بانزال علبة عصير الى الزبون وبذلك فان الحالة q1 تمثل حالة القبول او حالة النهاية للآلة. والمخطط التالي يمثل مخطط الحالات لآلة البيع.



ان DFAs هي آلات حالات منتهية محددة كما يدل عليها اسمها لكن يمكن لها أن تقبل عدد لانهائي من الكلمات. فآلات الحالات المنتهية المحددة DFA هي وسيلة محدودة شكلية لتمثيل لغات لانهائية infinite languages. و آلات الحالات المنتهية المحددة DFAs هي منتهية لانها تمتلك عدد محدود من الحالات finite states، ومن ثم ما يمكن أن تتذكره من معلومات حول الكلمات التي جرى قراءتها محدود بالضرورة. وتسمى اللغات التي تقبل من قبل هذه الآلات باللغات المنتظمة Regular Languages كما ذكرنا سابقا. وطبعا ليس كل لغة هي لغة منتظمة. وبعبارة أخرى هناك لغات لا يمكن قبولها من قبل آلات الحالات المنتهية المحدودة DFA او تمييزها بواسطتها. أنه ليست دائما توقع أو تنبؤ ما إذا كانت اللغة هي لغة منتظمة عملية واضحة. وكمثال على ذلك اللغة:

L={w | w contains as many subwords 01 as subwords 10}

هي لغة منتظمة لكن السؤال هنا هل بامكانك بناء آلة حالات منتهية محددة DFA لتمييزها؟. (تلميح: ليس عليك ان تتذكر كم خيط رمزي جزئي 01 او 10 جرت قراءته لكن كل ما عليك هو ملاحظة متى يتناوب الرمزان 0 و 1). وعلى العكس من هذا فانه من المستحيل بناء آلة حالات منتهية محدودة DFA للغة:

L= {w | w contains as many 0 as 1}

وهذه اللغة هي لغة ليست منتظمة لكنها لغة حرة السياق Context Free Language.

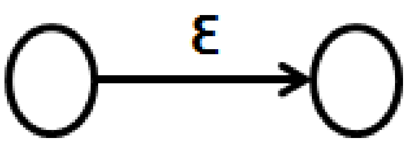
كما هو معلوم يخرج من كل حالة من حالات الآلة DFA عددا من الاسهم تمثل الانتقال من الحالة الحالية الى الحالة الجديدة. و يكون عدد الاسهم (الانتقالات) بعدد رموز الابجدية، اي سهم واحد (انتقالة واحدة) لكل رمز من رموز الابجدية. ومن ثم فان الانتقالة من الحالة الحالية الى اية حالة جديدة تكون معلومة اعتمادا على الرمز الذي جرت قراءته، لهذا سميت هذه الآلات بالمحددة Deterministic، اي يشير الى وحدانية الحل او الاحتساب Uniqueness of Computation . لكن ماذا يحدث لو كان الانتقال من حالة ما يؤدي الى اكثر من حالة جديدة اعتمادا على الرمز المقروء نفسه، أي أن هناك اكثر من احتمال للانتقال multiple transitions للرمز الواحد. كذلك ماذا يحدث اذا كان انتقال من حالة الى اخرى دون قراءة اي رمز من رموز الخيط الرمزي المراد التعرف عليه و يشار الى هذه الانتقالة بالرمز **ε**. والبعض الاخر من الحالات لا يوجد انتقال منها اعتمادا على بعض رموز الابجدية، اي يكون عدد الانتقالات منها اقل من عدد رموز الابجدية. فهل بامكان الآلة DFA أن تعالج مثل هكذا حالات؟. بالتأكيد الجواب سيكون لا تستطيع معالجة مثل هذه الحالات بسبب طبيعة تكوينها المحدد. لذا لابد من البحث عن حل لمعالجة الحالات الجديدة. والحل يكون باستعمال نوع جديد من الآلات يكون اسلوبه في الانتقال بين الحالات غير محدد Nondeterministic.

4.2 آلات الحالات المنتهية غير المحددة
Nondeterministic Finite State Machines(NFA)

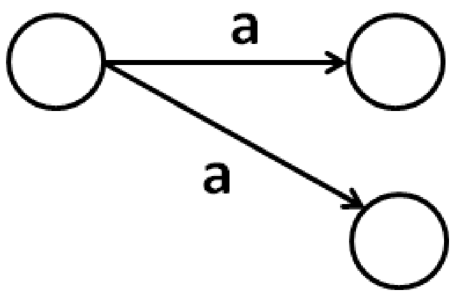
ظهرت هذه الآلات كحل للقصور الذي تعاني منه آلات الحالات المنتهية المحددة DFA في التعرف على بعض الخيوط الرمزية كما وضحنا سابقا. تسمى هذه الآلات بآلات الحالات المنتهية غير المحددة Nondeterministic Finite State Machines و يرمز لها بالرمز **NFA**. و تمتاز هذه الآلات بالصفات التالية :

(للتوضيح افترض ان الابجدية هي $\Sigma = \{a, b, c\}$).

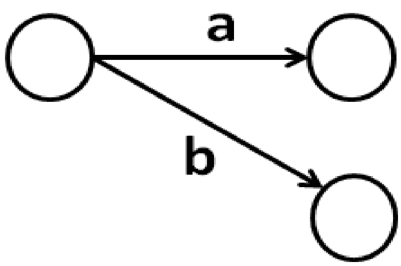
1. احتمال الانتقال من حالة الى اخرى دون قراءة رمز.



2. احتمالية الانتقال من حالة ما الى مجموعة من الحالات اعتمادا على قراءة رمز واحد من رموز الابجدية. اي تكون الآلة في حالات مختلفة في الوقت نفسه (بصورة متوازية).



3. عدد الانتقالات من حالة ما الى حالة اخرى قد يكون اقل من عدد رموز الابجدية، اي عدم وجود انتقالة من هذه الحالة الى حالة اخرى اعتمادا على رمز واحد او اكثر من رموز الابجدية. في الشكل ادناه لا يظهر انتقال اعتمادا على الرمز c.



يجري قبول الخيط الرمزي المدخل من قبل NFA اذا وصلت الى حالة القبول Accepting State و قد تصل الآلة الى حالة تسمى حالة الرفض dead end وعندها يجري رفض الخيط الرمزي من قبل الآلة.

تعرف آلة الحالات المنتهية غير المحددة NFA شكليا بانها الخماسي :

(K, ∑, δ, q0, F)

إذ أن:

1. K: تمثل مجموعة الحالات.
2. ∑: تمثل الابجدية.
3. q0: تمثل حالة البداية،إذ أن q0∈ K.

4. F: تمثل مجموعة حالات القبول، إذ أن $F \subseteq K$.

5. δ : دالة الانتقال و تعرف بالشكل:

$$\delta: K \times \Sigma^{\varepsilon} \rightarrow P(K)$$

وان $\Sigma_{\varepsilon} = \Sigma \cup \{\varepsilon\}$.

ملاحظات

1. الخيط الرمزي w يكون مقبولا من قبل الآلة NFA اذا كانت $\delta(q0, w)$ تحتوي على الاقل على حالة قبول واحدة او تسمى حالة نهاية End State.
2. اللغة التي تقبلها الآلة NFA هي مجموعة الخيوط الرمزية strings المقبولة من قبل هذه الآلة.

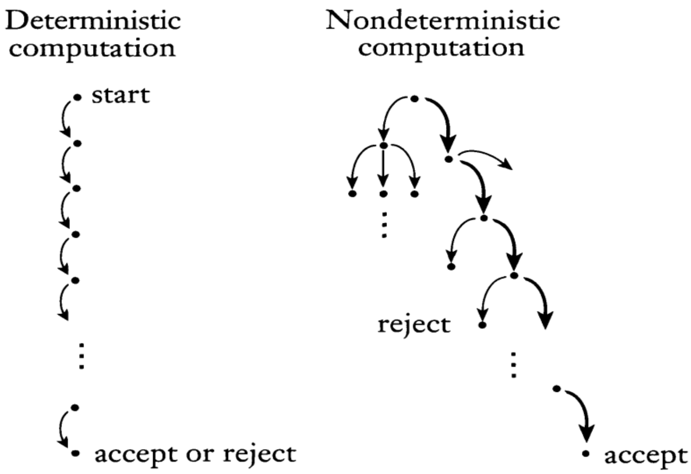
و تتميز NFA بما يلي:

1. سهولة البناء.
2. سهولة الفهم.
3. يمكن تحويل NFA الى DFA مكافيء لها.
4. صغيرة الحجم.

5.2 كيفية عمل آلة الحالات المنتهية غير المحددة NFA

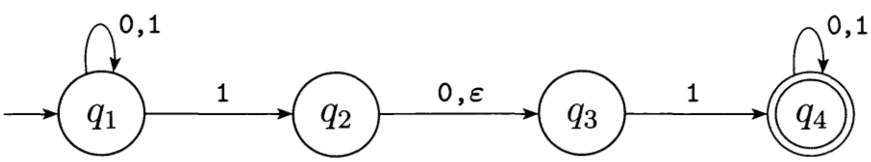
افترض اننا نريد تمييز احد الخيوط الرمزية string مثل w، ووصلنا الى حالة تؤدي الى مجموعة حالات، مثلا، اذا كانت الآلة الان في الحالة q1 و كان الرمز الذي جرت قراءته هو الرمز 1 وكان هناك اكثر من انتقالة واحدة من الحالة q1 اعتمادا على هذا الرمز فان الآلة في هذا الوضع تقوم بنسخ نفسها بعدد يساوي احتمالات الانتقال من تلك الحالة. ثم تقوم كل نسخة بالعمل منفردة لاكمال عملها وقد تستنسخ نفسها مرة اخرى اذا ظهر الوضع السابق نفسه. وفي النهاية اذا وجدت نسخة من هذه النسخ قد وصلت الى حالة النهاية (حالة القبول) فان الخيط الرمزي يكون مقبولا من الآلة و بعكسه فانه يرفض. إن هذا الوضع يفترض ان نسخ الآلة المتعددة تعمل بصورة متوازية في الوقت نفسه لايجاد الحل المناسب للمدخلات.

إن عمل NFA يشابه شجرة الاحتمالات tree of possibilities. يمثل جذرها بحالة البداية للالة وكل نقطة تفرع في الشجرة تقابل الحالة التي تكون الانتقالة فيها تؤدي الى اكثر من حالة واحدة. و تقبل الآلة الخيط الرمزي اذا انتهى على الاقل احد الفروع بحالة قبول Accepting state. كما في الشكل (2-5).



الشكل (2-5) تمثيل NFA بشكل شجرة

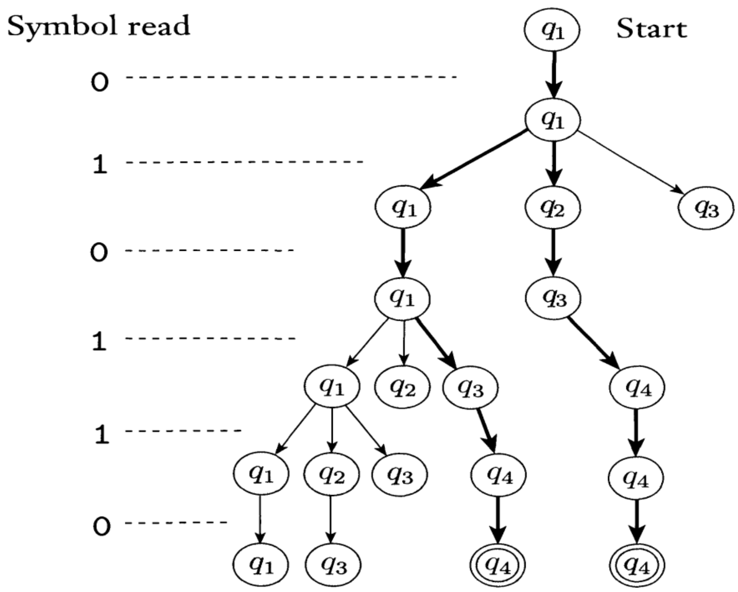
تأمل الشكل التالي الذي يمثل NFA



اذا كانت الآلة NFA اعلاه تعالج الخيط الرمزي التالي 010110 فان شجرة الاحتمالات التي تمثل عملها تكون بالشكل (2-6):

الشرح:

في البداية تكون الآلة في حالة البداية في q1.



الشكل(2-6) شجرة الاحتمالات للالة

1. عند قراءة الرمز 0 تكون الانتقالة الى الحالة نفسها اي تبقى الآلة في الحالة q1 .
2. عند قراءة الرمز 1 هناك الاحتمالات التالية:
   a. اما تبقى الآلة في نفس الحالة الحالية اي q1 ومن ثم فان هناك نسخة للآلة اعتمادا على هذا الاحتمال و يمثله الفرع من جهة اليسار و من ثم تستكمل الآلة عملها في هذا الفرع حتى النهاية.
   b. تنتقل الآلة الى الحالة q1 و تنسخ لها نسخة جديدة يمثلها الفرع الاوسط و من ثم تستكمل الآلة عملها في هذا الفرع حتى النهاية.
   c. او عند انتقال الآلة الى الحالة q2 فانها تستمر بالانتقال الى الحالة q3 دون قراءة اي رمز و يمثلها الفرع الايمن والذي تتوقف فيه الآلة معلنة رفض الرمز في هذه النسخة.
3. الفرع الاوسط يستمر بعمله و لايواجه اي تفرعات جديدة و ينتهي الى حالة قبول للخيط الرمزي.
4. الفرع الايسر يواجه تفرعا كما حصل في النقطة 2 وانتهت التفرعات الى رفض الخيط الرمزي ماعدا فرعا واحدا انتهى الى حالة قبول.

مثال7

صمم الة حالات منتهية غير محددة NFA تميز اللغة التي جميع كلماتها تنتهي بالخيط الرمزي الجزئي 01.

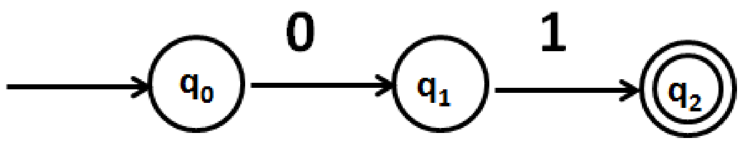
L=**{w | w ends with subword 01}**

الحل

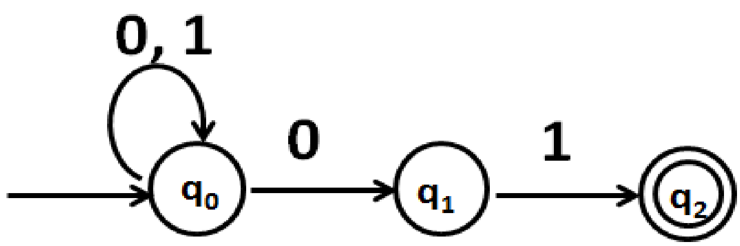
لتصميم اللغة ... نجد أن الصفة المميزة لها أن كل كلمة فيها تنتهي ب 01.

إذن أية كلمة مهما كان طولها و طبيعة تسلسل الرموز فيها فهي تكون بالشكل w01.

لتصميمها نبدأ اولا بالصفة و نحدد عدد الحالات والانتقالات التي تميز الخيط الرمزي المميز و هو 01. نجد اننا نحتاج الى ثلاث حالات :الاولى للبدأ بعملية القراءة، والثانية للانتقال من الحالة الاولى اليها عند قراءة الرمز 0 بينما الثالثة تمثل حالة القبول عند الانتقال من الحالة الثانية اليها بعد قراءة الرمز 1. والشكل التالي يوضح هذه العملية.



هذه الآلة تقبل الخيط الرمزي 01 فقط. ولجعل الآلة تقبل جميع الخيوط الرمزية التي تنتهي بالخيط الرمزي الجزئي 01 فان على الآلة ان تقرأ جميع الرموز مهما كان تسلسلها عند الحالة q_0 دون الانتقال الى أية حالة جديدة، اي تبقى عند الحالة نفسها. لذا نضيف سهما ينطلق من الحالة q_0 و يعود اليها مؤشرا بالرمزين 0 و 1. والشكل التالي يوضح شكل الآلة النهائي.



و تعرف هذه الآلة NFA شكليا كما يلي: **({q0, q1, q2}, {0, 1}, δ, q0, {q2})** إذ أن δ تعرف بالجدول التالي:

	0	1
→ q0	{q0, q1}	{q0}
q1	ϕ	{q2}
q2	ϕ	ϕ

و لتوضيح عمل الآلة NFA المميزة للغة المعطاة سوف نبني شجرة الاحتمالات. اول افتراض سوف نأخذه ان الكلمة المدخلة هي 01

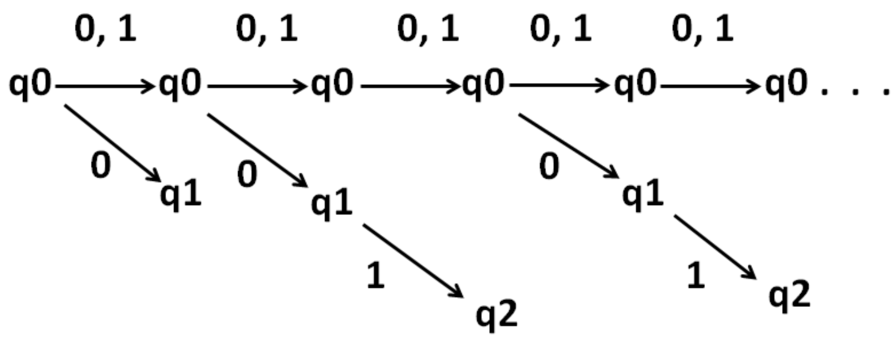
- في هذا الوضع سوف تبدأ الآلة بحالة البداية start state و سوف نسميها q_0. عند قراءة الرمز 0 سوف تنتقل الآلة الى حالة جديدة هي الحالة q_1. و عندما تقرأ الرمز الجديد 1 فانها سوف تنتقل الى الحالة الجديدة q_2 و التي عندها تنتهي الآلة من عملية القراءة و تقبل الكلمة المدخلة لذا نسمي q_2 بحالة القبول accepting state.

- طبعا ان اي كلمة بطول رمز واحد هي كلمة مرفوضة من قبل الآلة لانها لا تحقق الشرط وهو انتهاء الكلمة بالخيط الرمزي الجزئي 01. لذا اذا كان الرمز المدخل 0 سوف يتوقف عند الحالة q_0 واذا كانت قيمته 1 فيجب ان تكون هناك انتقالة الى حالة ترفض فيه هذا الرمز وهنا اما ان نضيف حالة جديدة مثل q_3 او نكتفي بان تكون الانتقالة الى الحالة نفسها اي تبقى الآلة في الحالة q_0.

- اذا كانت الكلمة المدخلة هي 00 او 11 او 10 : عندما تكون الكلمة 00 فان الآلة تنتقل من الحالة q_0 الى الحالة q_1 عند قراءة الرمز 0 وعند قراءة الرمز الثاني 0 فان الآلة لا تستطيع ان تنتقل الى حالة جديدة لذا لحل هذا الاشكال يجب عدم الانتقال الى الحالة q_1 وانما اما الانتقال الى حالة جديدة مثل q_3 او البقاء عند الحالة q_0 وهذا هو الاختيار الافضل حتى لا يكبر حجم الآلة. و بهذا يكون الانتقال من الحالة q_0 الى نفسها يكون عند قراءة الرمز 0 وكذلك عند قراءة الرمز 1 كما

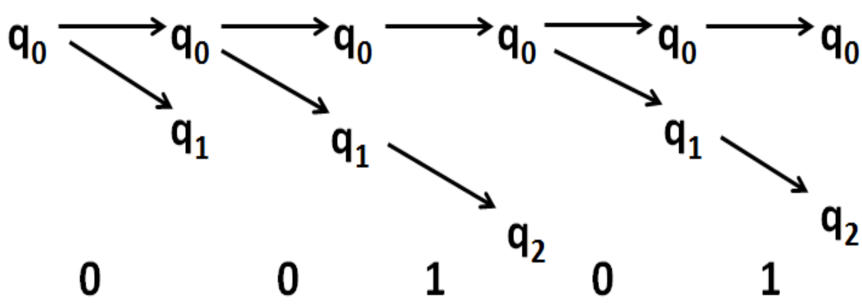
وضحنا في النقطة السابقة. والان اصبحت الامور واضحة للكلمتين 11 و 10 إذ أنهما سوف يبقيان عند الحالة q_0 اي ترفض الكلمتان لعدم توفر الشرط.

- اما اي كلمة طولها اطول من رمزين فاذا كانت لا تنتهي بالخيط الرمزي الجزئي 01 سوف ترفض في احدى حالات الرفض اما اذا كانت تحتوي عليه فانها سوف تصل الى حالة القبول q_2 كما يوضحه شكل شجرة الاحتمالات التالي:



طبعا لا ننسى ان الآلة تنسخ نفسها عند كل تفرع بعدد يساوي عدد التفرعات و تشتغل كل النسخ لتعمل بصورة متوازية لايجاد الجواب المطلوب.

وكمثال افترض ان الكلمة المدخلة هي 00101 فشجرة الاحتمالات تكون كما يلي:



التعريف الشكلي لعمل آلة الحالات المنتهية غير المحددة NFA :

لتكن $N = (K, \sum, \delta, q_0, F)$ هي الآلة NFA وان w تمثل الكلمة المعرفة على الابجدية $\sum$. يقال أن N تقبل الكلمة w اذا استطعنا كتابة w بالشكل $w=y_1y_2\dots y_m$ **إذ أن** y_i رمز ينتمي الى الابجدية $\varepsilon \sum$ و سلسلة الحالات $r_0, r_1, \dots, r_m$ موجود ضمن المجموعة K تحت الشروط الثلاثة التالية:

1. $r_0 = q_0$,

2. $r_{i+1} \in \delta(r_i, y_{i+1})$, for i = 0, ..., m - 1, and

3. $r_m \in F$.

الشرط الاول: تبدأ الآلة عند حالة البداية Start State.

الشرط الثاني: ان الحالة ri+1 هي احدى الحالات المسموح للالة N ان تنتقل اليها عندما تكون موجودة في الحالة ri و عند قراءة الرمز yi+1 . لاحظ بان δ (ri, yi+1) تمثل مجموعة الحالات التالية المسموح بها و لهذا نقول بان الحالة ri+1 هي عنصر ينتمي الى هذه المجموعة.

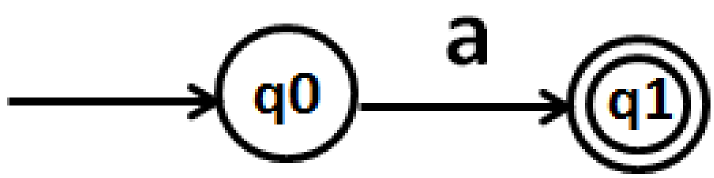
الشرط الثالث: يقال بان الآلة N تقبل الكلمة المدخلة w اذا كانت اخر حالة تصل اليها الآلة هي حالة القبول Accepting State.

مثال8

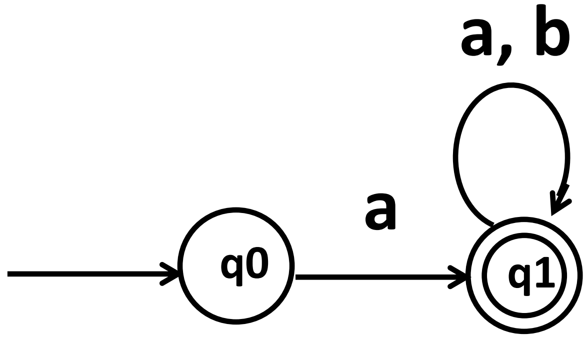
صمم آلة NFA لتقبل لغة كلماتها تبدأ بالرمز a معرفة على الابجدية {a, b}.

الحل

ان جميع كلمات هذه اللغة تكون بالصيغة aw إذ أن w خيط رمزي متكون من خليط من الرموز a و b وباي ترتيب. إن المهم هنا اول رمز من الخيط الرمزي، فلو فرضنا ان البخط الرمزي المدخل هو a فان الآلة ي عندها تبدأ بحالة البداية مثل q0 التي عندها تقرأ الرمز a و تنتقل الى الحالة الجديدة مثل الحالة q1 التي عندها تقبل الخيط الرمزي المدخل وتمثل بذلك حالة القبول accepting state . يكون شكل الآلة الاولي كما يلي:



الان اذا كان الخيط الرمزي طوله اكبر من واحد فتقوم الآلة بقراءة رموز الجزء w في الحالة q1 لحين انتهاء الخيط الرمزي ويصبح شكل الآلة النهائي بالشكل التالي:



مثال9

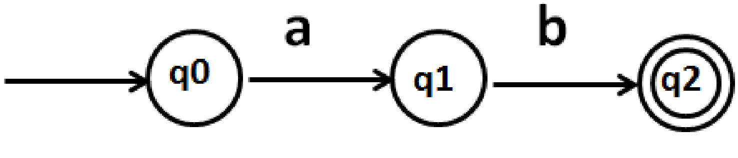
صمم الة NFA تقبل كل الخيوط الرمزية التي تبدأ بالرمز a و تنتهي بالرمز b.

الحل

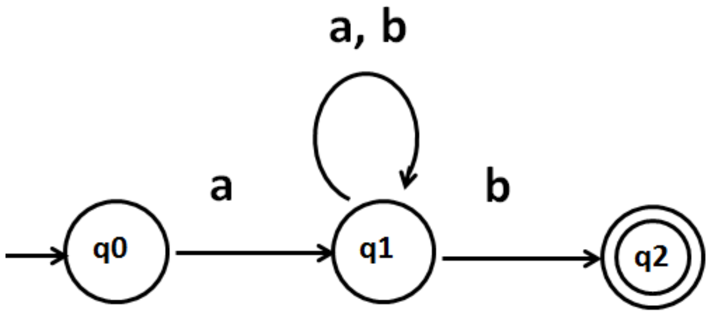
إن الصفة المميزة للخيوط الرمزية هنا انها تبدأ بالرمز a و تنتهي بالرمز b. لذا فان الحالة التي تبدأ فيها الآلة NFA هي الحالة التي تقرأ فيها الرمز a بينما الحالة التي تنتهي فيها فهي الحالة التي يكون فيها الرمز المقروء هو الرمز b وتمثل حالة القبول. فلتصميم الآلة NFA نختار حالتين: الاولى حالة البداية، والثانية حالة القبول.

طبعا هنا اي خيط رمزي طوله اقل من 2 ترفضه الآلة مباشرة. اما اذا كان طول الخيط الرمزي اكبر من او يساوي 2 فيكون وضع الآلة بما يلي:

عندما يكون طول الخيط الرمزي يساوي 2 فهناك الاحتمالات التالية: aa, ab, ba, bb. بالنسبة للخيط الرمزي ab، تقوم الآلة بقراءة الرمز الاول عند حالة البداية ، q0، و تنتقل الى الحالة الجديدة مثل حالة q1. عند الحالة q1 تقوم الآلة بقراءة الرمز الثاني b و تنتقل الى الحالة الجديدة مثل q2 التي تقبل الخيط الرمزي ويكون شكل الآلة كما يلي:



أما اذا كان الخيط الرمزي هو bb او ba فان الآلة ترفضه مباشرة عند حالة البداية لانه لا يبدأ الرمز a. اما الخيط الرمزي aa فان الآلة ترفضه عند الحالة q1 لانه لا ينتهي بالرمز b. وعندما يكون الخيط الرمزي طوله اكبر من 2 فيجب ان يكون بالصيغة awb إذ أن w عبارة عن سلسلة من رموز a و/او b. ومن ثم فان رموز الخيط الرمزي w تقرأ من قبل الآلة عند الحالة q1 دون الانتقال الى حالة جديدة و يصبح شكل الآلة النهائي كما يلي:



6.2 الفرق بين DFA و NFA

كل حالة في DFA لها بالضبط مخرج واحد لكل رمز ينتمي الى الابجدية يمثل الانتقالة الى حالة اخرى. اما في NFA فقد يوجد مخرج واحد او اكثر لكل رمز ينتمي الى الابجدية وقد لا يوجد اي مخرج.

في DFA، توجد علامة دالة على كل سهم لانتقالة تدل على رمز الابجدية الذي يجري الانتقالة بموجبه. في NFA، على العموم، قد توجد علامة دالة تشير الى رمز من رموز الابجدية او تشير الى الرمز ε الذي يدل على الانتقال دون قراءة اي رمز. وقد يوجد سهم او اكثر معلم بهذه العلامة او قد لا يوجد مثل هذا السهم.

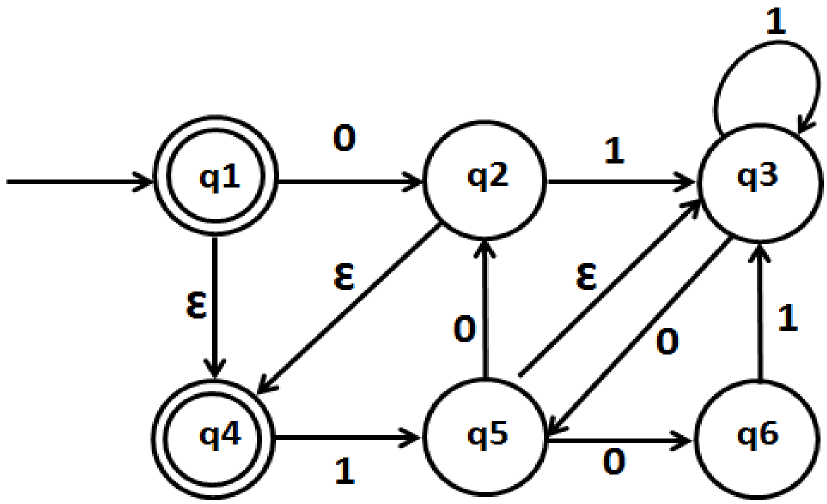
7.2 تكافؤ DFA و NFA

تميز آلات الحالات المنتهية بنوعيها المحددة Deterministic و آلات الحالات المنتهية غير المحددة Nondeterministic الصنف نفسه من اللغات. و مثل هذا التكافؤ يكون مدهشا و مفيدا. يكون مدهشا لان NFA تظهر بانها اكثر قوة من DFA، بحيث يتوقع ان NFA ستميز اكثر من هذه اللغات. و مفيدة لان NFA احيانا تفسر اللغة المعطاة بطريقة ابسط مما تفعله DFA مع اللغة نفسها احيانا.

- يقال ان التين تكون متكافئة اذا ميزت اللغة نفسها.

❖ كل NFA لها DFA مكافئة.

افترض ان N هي الآلة NFA معرفة على مجموعة الابجدية ∑ و مجموعة الحالات K. و لتكن حالة البداية هي الحالة q_1. ولكي نأخذ مثالا يساعدنا في توضيح الآلة، دعنا نفرض ان N معرفة كما يلي:

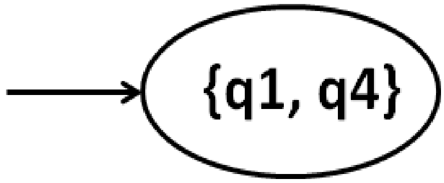


هدفنا هو بناء الآلة DFA والتي نسميها M بحيث L(M)=L(N).

الفكرة الرئيسة: ان M سوف تحتاج الى تتبع كل الحالات الممكنة التي كانت فيها الآلة N بعد كل سلسلة ممكنة من الرموز. لكي نعمل ذلك، ستكون حالات M متوافقة مع مجموعات جزئية من حالات N.

الخطوة الاولى: تحديد حالة البداية للآلة M.

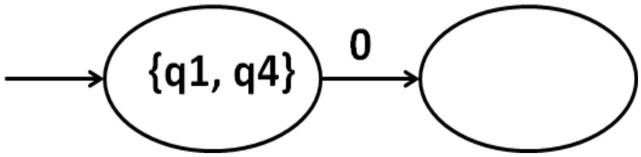
ارسم شكلا بيضويا ليمثل حالة البداية للآلة M. ثم حدد مجموعة الحالات الجزئية من N لتشير الى حالة البداية في M. في مثالنا، ان مجموعة الحالات الجزئية تتكون من حالة البداية q_1 وكل الحالات التي يمكن الانتقال اليها من q_1 دون قراءة اي رمز اي الإنتقالة ε. وفي مثالنا تكون لدينا المجموعة الجزئية $\{q_1, q_4\}$ تمثل مجموعة كل الحالات في N الممكنة التي لا يحصل فيها قراءة اي رمز. لذا تكون حالة البداية للآلة M كما في الشكل التالي:



الخطوات الوسطية: تحتاج الى تكرار هذه الخطوة حتى اكتمال المخطط M الذي يمثل الآلة DFA.

خذ اي رمز **σ** من الابجدية ∑ واي حالة في المخطط M بحيث أن هذه الحالة تفتقد للانتقالة المؤشرة بالرمز **σ**. ارسم سهم يمثل الانتقالة من الحالة التي اخترتها الى حالة جديدة و علمها بالعلامة **σ**.

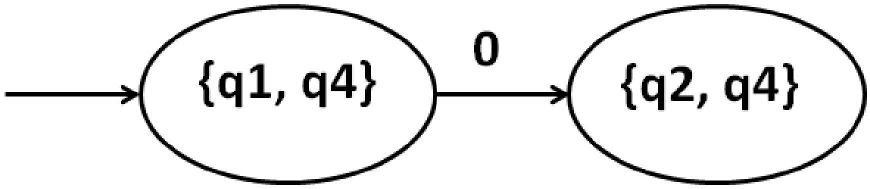
في مثالنا، لدينا حالة واحدة فقط لحد الان في M: لا توجد انتقالة تؤدي الى الخروج منها، لنؤشرها بالعلامة 0.



نحتاج الان لاعطاء الحالة الجديدة اسم يدل عليها كما فعلنا مع حالة البداية، لذا نحتاج الى تحديد مجموعة الحالات التي تمثلها.

ان مجموعة الحالات التي نريد ان نسمي بها الحالة الجديدة تتضمن كل الحالات في N التي يمكن الوصول الى الحالة الجديدة منها عن طريق قراءة الرمز σ مع اي عدد من الانتقالات Ɛ (عدد اكبر من او يساوي 0).

في مثالنا، هذا يعني ان الحالة الجديدة ستعطى اسما يمثل بمجموعة من الحالات في N بحيث تنتقل اليها اما من الحالة q_1 او الحالة q_4 و عند قراءة الرمز 0 مع اي عدد من الانتقالات Ɛ. و بعد دراسة المخطط N نجد ان المجموعة $\{q_2, q_4\}$ هي المجموعة التي سنسمي بها الحالة الجديدة.

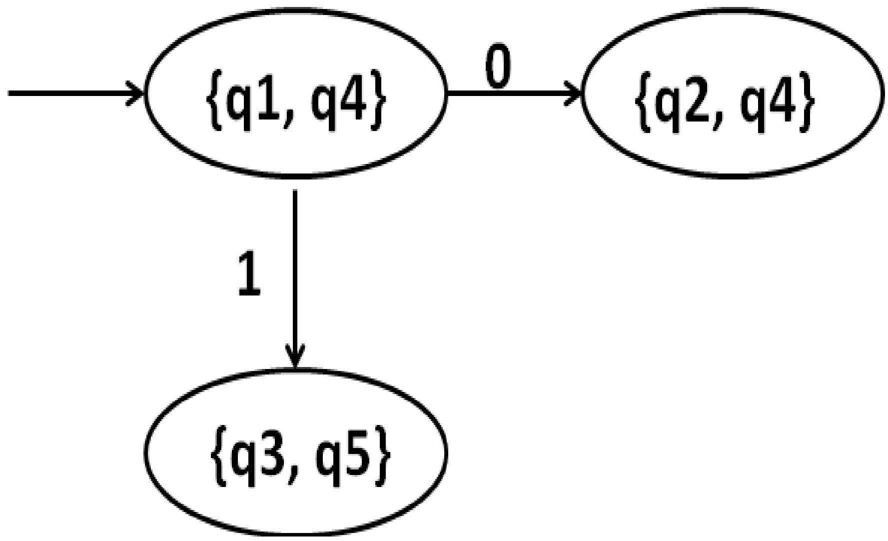


ملاحظة

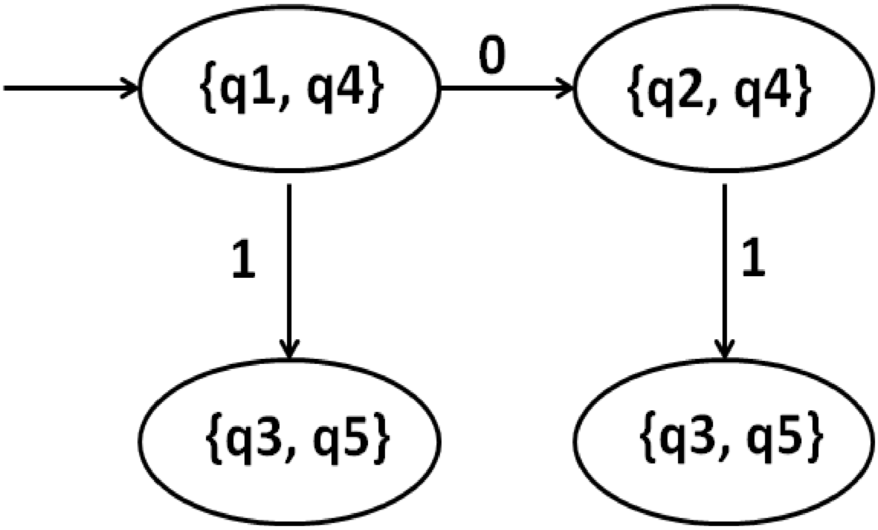
اذا كانت المجموعة التي حصلنا عليها قد سميت بها احدى الحالات في المخطط M فيما سبق فيجري حذف الحالة الجديدة و ترسم انتقالة الى الحالة التي تحمل هذا الاسم(مجموعة الحالات).

الان نستمر بالعمل السابق نفسه مع حالة البداية $\{q_1, q_4\}$ لكن مع الرمز 1. سوف نضع حالة جديدة و نسميها بمجموعة الحالات المناسبة حسب المخطط N. و هذه المجموعة هي $\{q_3, q_5\}$.

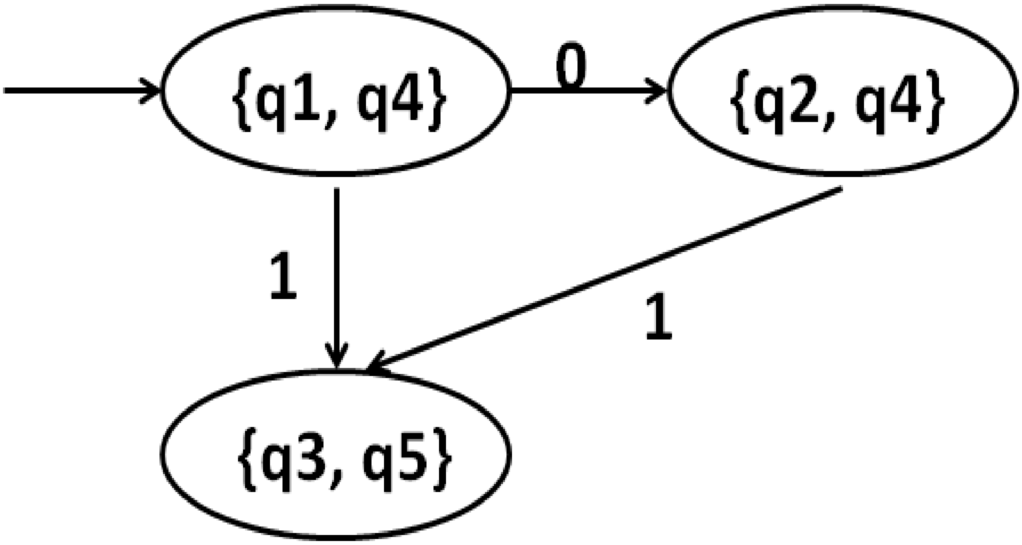
وللتوضيح هناك انتقالة من الحالة q_4 الى q_5 عند قراءة الرمز 1 و كذلك هناك نفس الانتقالة السابقة الا انه يليها انتقالة من الحالة q_5 الى الحالة q_3 دون قراءة اي رمز، اي الانتقالة Ɛ. و بذلك تكون مجموعة الحالات هي المجموعة $\{q_3, q_5\}$.



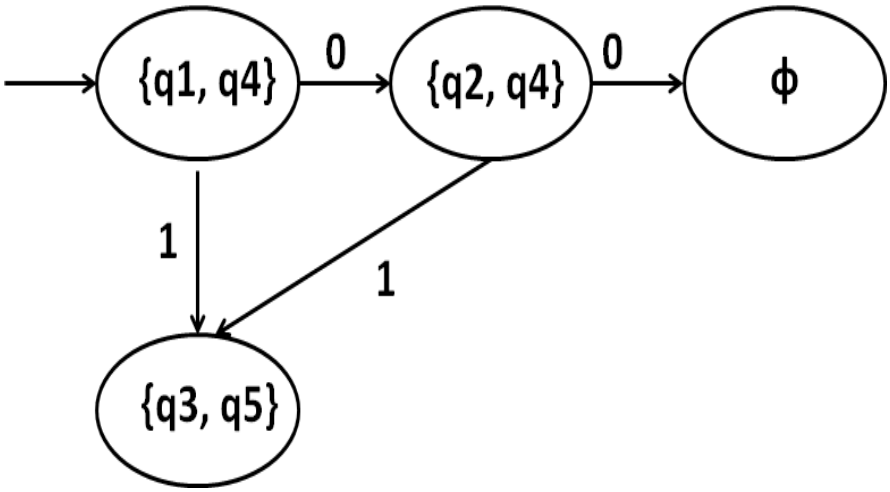
الان لو اخذنا الحالة $\{q_2, q_4\}$ مع الرمز 1. فان الانتقالة الى الحالة الجديدة بعد دراسة المخطط N تكون بالشكل التالي:



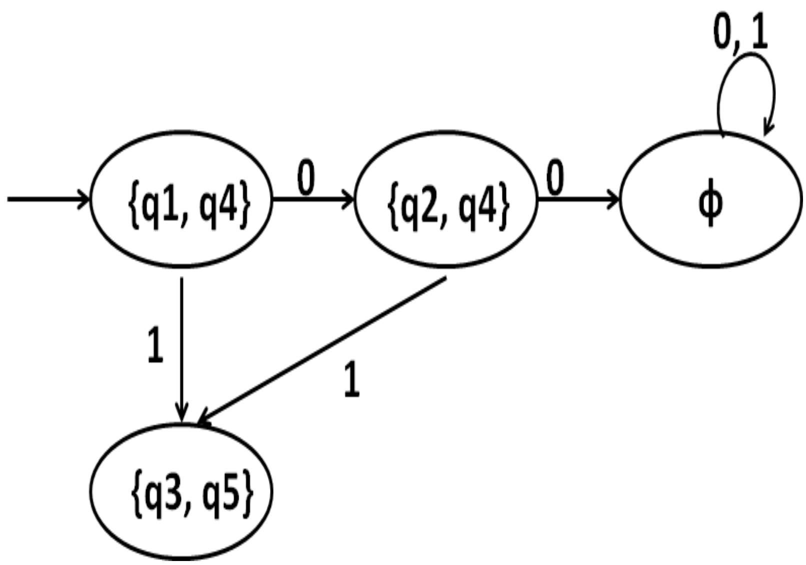
نجد ان الحالة الجديدة مشابهة لحالة سابقة، لذا و حسب الملاحظة السابقة يجري حذف هذه الحالة و رسم سهم الى الحالة المشابهة مؤشرة بالرمز 1. والشكل التالي يوضح هذا التبسيط.



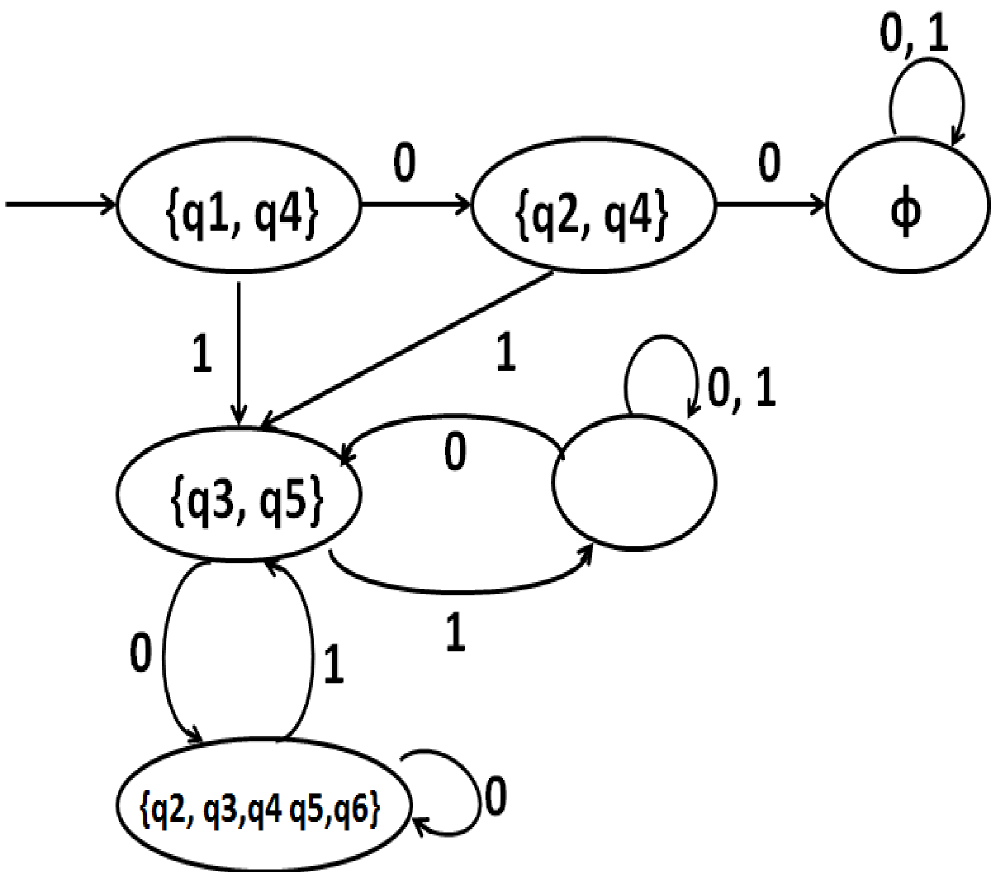
احيانا قد تحصل على مجموعة فارغة ϕ تعامل معها كاي مجموعة اخرى و سمي الحالة الجديدة بها. مثلا، عند الانتقال من الحالة $\{q_2, q_4\}$ مع الرمز 0 فاننا نحصل بعد دراسة المخطط Nعلى الشكل التالي:



ان السبب في الحصول على مجموعة خالية لانه لا يوجد في المخطط N انتقالات من الحالة q_2 اوالحالة q_4 الى حالة اخرى عند قراءة الرمز 0. ومن السهل جدا التعامل مع الانتقالات الخارجة من الحالة المؤشرة بالمجموعة الفارغة ϕ فتمثل باسهم تشير الى انتقالات الى الحالة نفسها مؤشرة في جميع رموز الابجدية إذ لا يوجد حالات ينتقل اليها.



وبالاستمرار بهذه الخطوة نفسها على بقية الحالات نحصل على المخطط التالي:

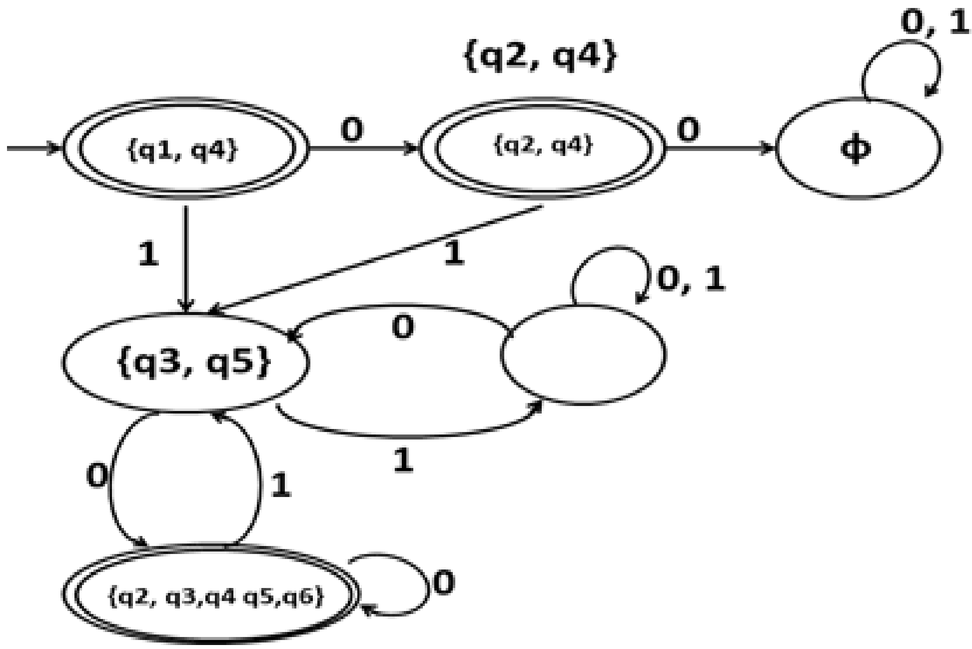


و بعد ان نتأكد من ان جميع الحالات تخرج منها اسهم بعدد رموز الابجدية ننتقل الى الخطوة التالية.

الخطوة الاخيرة: بقي ان نحدد من هي الحالات في المخطط M التي تقبل الكلمة المدخلة.

- كل حالة تعد حالة قبول فقط واذا فقط سميت بمجموعة حالات تتضمن حالة القبول في المخطط N.

في مثالنا، ان حالة القبول في المخطط N هي الحالة q_1 والحالة q_4، لذا اي مجموعة حالات في M تنتمي اليها q_1 او q_4 فانها تمثل حالة قبول في المخطط M. و بذلك تكون الصيغة النهائية لمخطط الآلة DFA المكافئة الى NFA في مثالنا كما في الشكل التالي:



و للتأكد من صحة عملية التحويل تختبر الآلة DFA باستعمال عدد من الكلمات التي تنتمي الى اللغة نفسها. ان عدد الحالات في الآلة DFA في هذا المثال قليل لكن علينا معرفة ان عدد الحالات يتزايد أسيا اعتمادا على عدد حالات الآلة NFA.

8.2 تصميم DFA لتمييز لغة متولدة من اتحاد لغتين
Design a DFA through the union of two other DFAs

المسألة: اذا اردت تصميم DFA لتمييز لغة معرفة كاتحاد لغتين منتظمتين Regular Languages، مثلا، تمييز جميع الكلمات المعرفة على الابجدية {0, 1} بحيث اما تبدأ بالرمز 0 او تنتهي بالرمز 0 ،يجب اتباع الخطوات التالية.

1 -بناء DFA لكل لغة جزئية على حدة.
2 -ربط الالتين DFA1 و DFA2 بواسطة حالة بداية جديدة اضافية و انشاء انتقالة ε من هذه الحالة الجديدة الى حالة البداية في كل من الالتين DFA1 و DFA2 و بذلك تحصل على الة حالات منتهية غير محددة NFA لاحتواء الآلة الجديدة على الانتقالات من نوع ε. تقبل الآلة الجديدة NFA اللغة المتولدة من اتحاد لغتين و ما عليك سوى تحويل الآلة NFA الى الآلة DFA كما جرى شرحها سابقا.

3 -تبسيط (minimize) الآلة DFA عند الضرورة.

مثال10

صمم الة DFA لتمييز اللغة L المعرفة بالصيغة التالية:

L={w: w over {0, 1} and w either begins with 0 **or** ends with 0}

تتكون اللغة L من اتحاد لغتين هما:

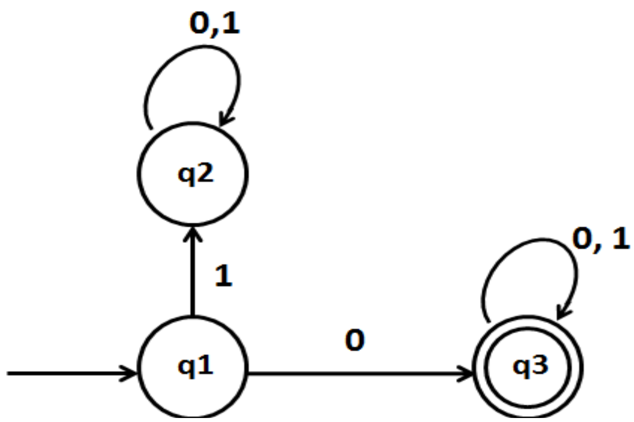
L_1={w: w over {0, 1} and w begins with 0}

و

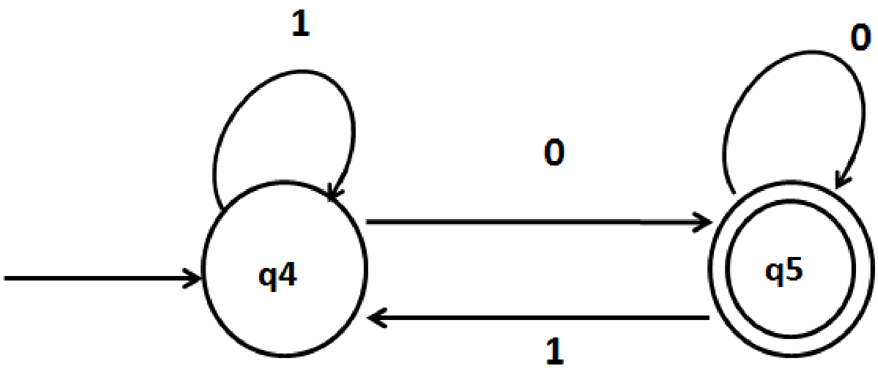
L_2={w: w over {0, 1} and w ends with 0}

الحل

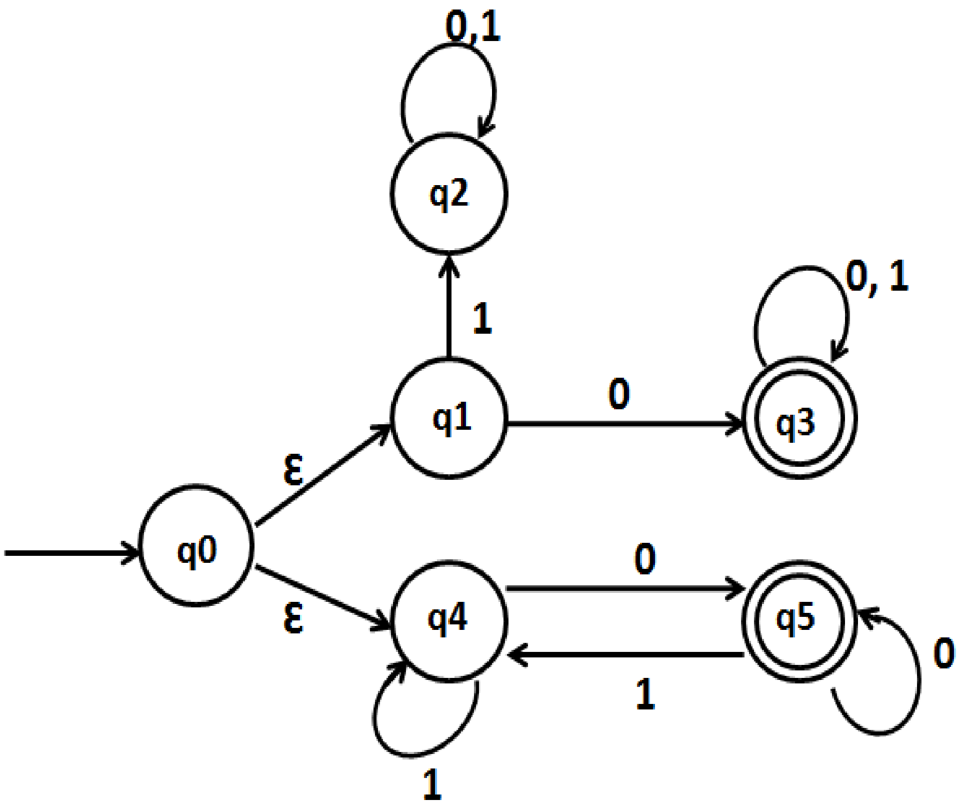
1. نصمم الآلة DFA1 لتمييز اللغة L_1.



2. نصمم الآلة DFA2 لتمييز اللغة L_2.

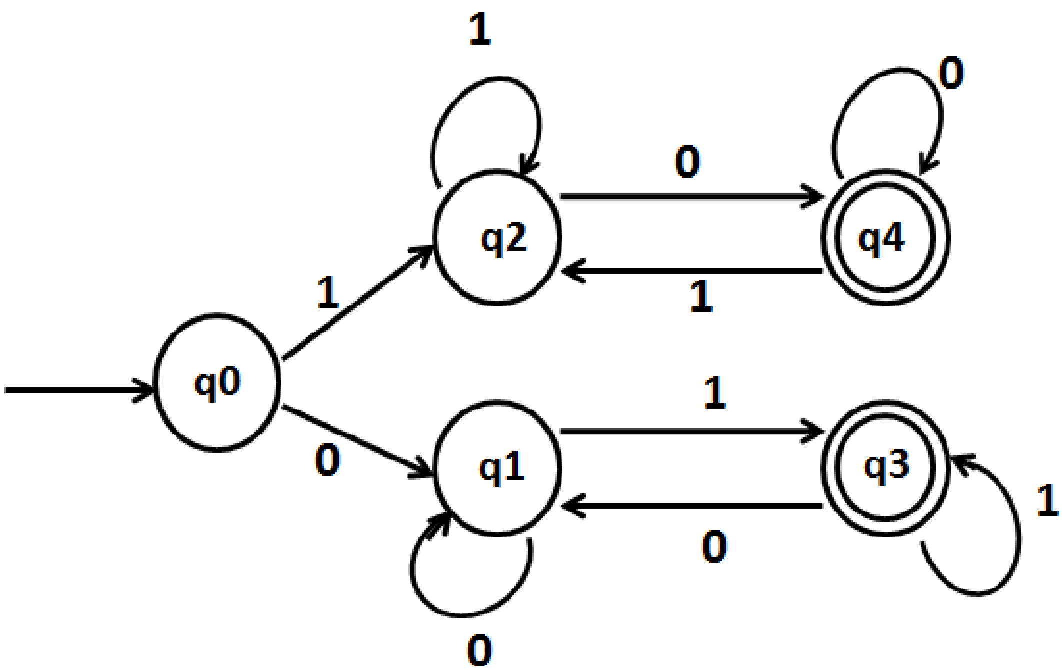


3. نربط الالتين عن طريق إضافة حالة جديدة وعدها حالة بداية للآلة الجديدة المتولدة من ربط الالتين بحيث يخرج منها سهمان احدهما يرتبط بحالة البداية للآلة الاولى والسهم الثاني يربط بحالة البداية للآلة الثانية. و يؤشر كل سهم بالرمز ε.



وبعد التحويل من NFA الى DFA حسب الخطوات التي جرى شرحها سابقا نحصل على الآلة التالية:

إذ أن الحالة q_0 تقابل المجموعة $\{0, 1, 4\}$.
والحالة q_1 تقابل المجموعة $\{3, 5\}$.
والحالة q_2 تقابل المجموعة $\{2, 4\}$.
والحالة q_3 تقابل المجموعة $\{3, 4\}$.
والحالة q_4 تقابل المجموعة $\{2, 5\}$.



9.2 تصميم DFA من خلال متمم آلة DFA اخرى.
Design a DFA through the Complementation of another DFA

المسألة: بناء الة حالات منتهية محددة DFA تقوم بتمييز لغة معرفة كمتمم للغة منتظمة Regular Language او محتوياته موصوفة بكلمات مثل not او no او none. مثلا، اللغة المعرفة بالصيغة:

L= { w | w is not a multiple of 3 when interpreted as a binary integer}

الحل

للحصول على مثل هذه الآلة:

1. نقوم بتصميم الآلة DFA لتمييز اللغة L (نسميها اللغة الموجبة Positive Language).
2. نحول كل حالة قبول Accepting State الى حالة غير قبول Non Accepting State وبالعكس لجميع حالات الآلة DFA.
3. الآلة المتولدة من حالة التحويل هذه تميز متمم اللغة L وهي الآلة المطلوبة و تقبل كل الكلمات التي ترفضها اللغة L.

مثال11

صمم الة DFA لتمييز اللغة L المعرفة بالصيغة:

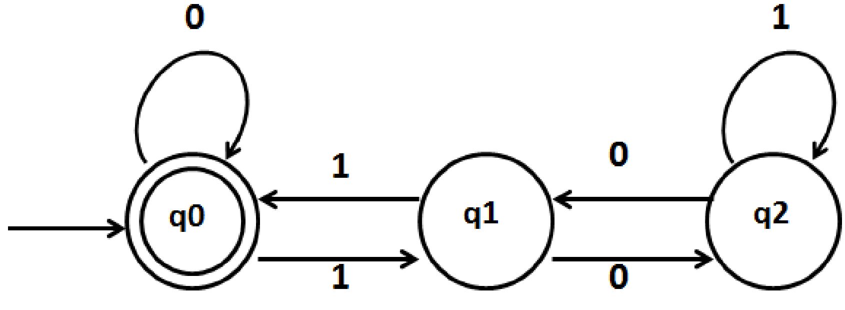
L = { w | w is *not* a multiple of 3 when interpreted as a binary integer}

بحيث أن اللغة موجبة لها هي L_1 المعرفة بالصيغة:

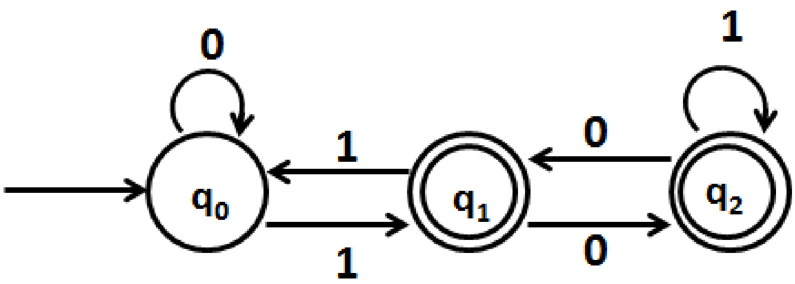
L_1 = { w | w is a multiple of 3 when interpreted as a binary integer}

الحل

الآلة DFA المميزة للغة L_1 هي كما في الشكل التالي وقد جرى تصميمها فيما سبق.



اما الآلة DFA المميزة للغة L معرفة بالمخطط التالي بعد قلب الحالات .



10.2 تصميم DFA لتمييز لغة متولدة من تقاطع لغتين منتظمتين
Design a DFA that accepts the intersection of two regular languages

المسألة: إذا اردت تصميم DFA لتمييز لغة معرفة كتقاطع لغتين منتظمتين Regular Languages او وصف محتوياتها يتضمن الكلمة and، مثلا، تمييز جميع الكلمات المعرفة على الابجدية {0, 1} بحيث اما تبدأ بالرمز 0 او تنتهي بالرمز 0، يجب اتباع الخطوات التالية.

الحل

يعتمد الحل على **قانون دي مورغن De Morgan's law**. اذا كانت لديك اللغتان L1 و L2، فحسب قانون دي مورغن يعرف التقاطع بدلالة الاتحاد و المتمم:

$$\mathbf{L_1 \cap L_2 = complement\ (\ complement(L_1) \cup complement(L_2))}$$

اي

$$\mathbf{L_1 \cap L_2 = (L_1^c \cup L_2^c)^c}$$

لبناء DFA للمسألة نتبع الخطوات التالية:

1. نصمم DFA1 للغة L1 و DFA2 للغة L2 كما جرى في مسألة اتحاد اللغتين.
2. نطبق عملية المتمم على الالتين لنحصل على آلتي DFA جديدة تمثل متمم اللغتين نسميهما CDFA1 و CDFA2 .
3. نربط الالتين بعملية or لنحصل NFA.

4. نقوم بتحويل NFA الى DFA.
5. نطبق عملية المتمم مرة اخرى على DFA الناتجة من خطوة (4) لنحصل على آلة DFA تميز تقاطع لغتين منتظمتين.
6. بسط المخطط ان احتاج الى تبسيط.

مثال 12

صمم الة DFA لتمييز اللغة L المعرفة بالصيغة التالية:

L={w: w over {0, 1} and w either begins with 0 **and** ends with 0}

تتكون اللغة L من اتحاد لغتين هما:

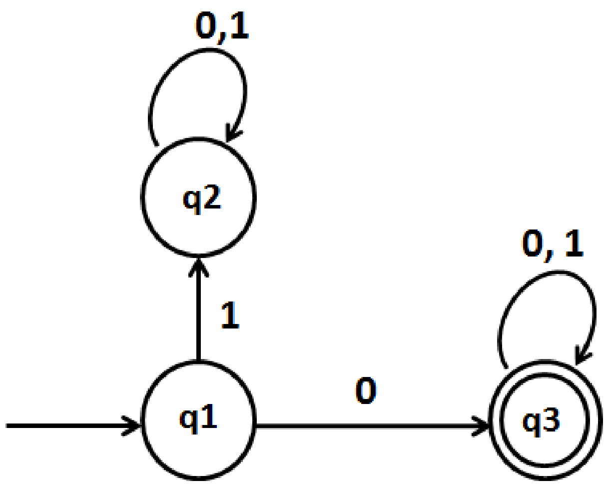
L_1={w: w over {0, 1} and w begins with 0}

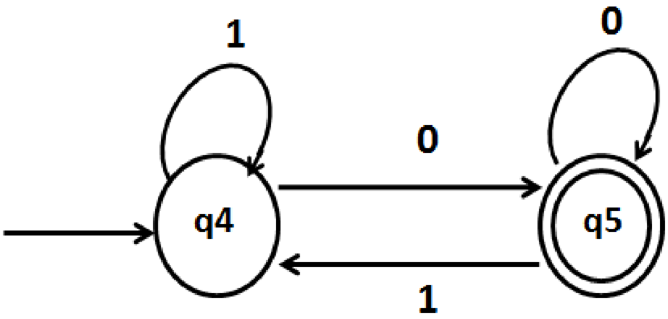
و

L_2={w: w over {0, 1} and w ends with 0}

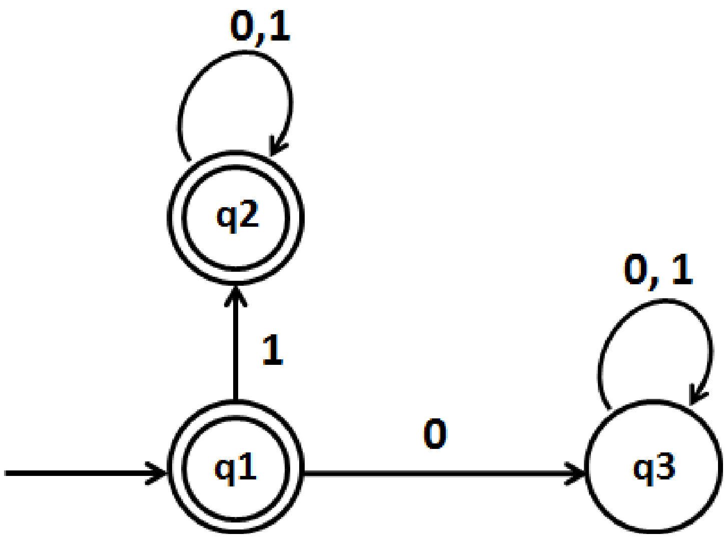
الحل

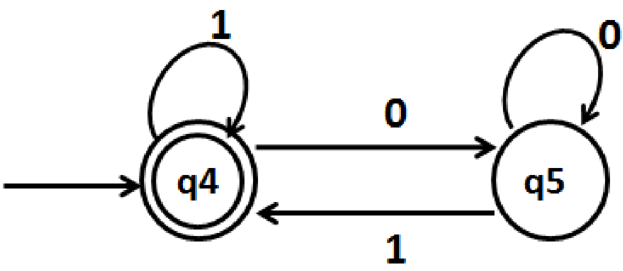
1. نصمم DFA1 للغة L_1 و DFA_2 للغة L_2 كما جرى في مسألة اتحاد اللغتين.



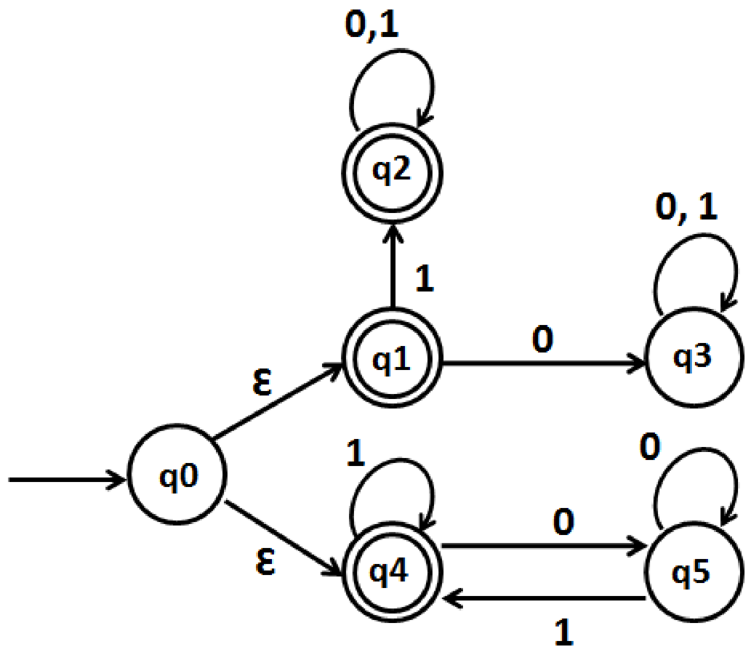


2. نطبق عملية المتمم على الالتين لنحصل على آلتي DFA جديدة تمثل متمم اللغتين نسميهما $CDFA_1$ و $CDFA_2$.

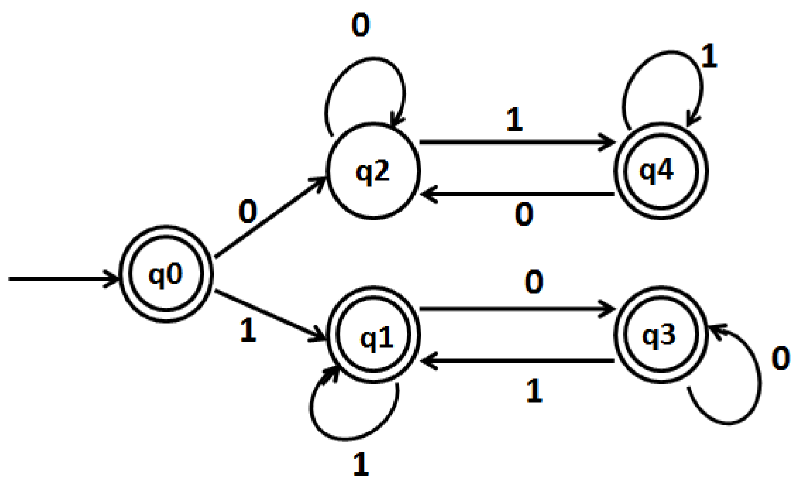




3. نربط الآلتين بعملية or لنحصل على NFA.



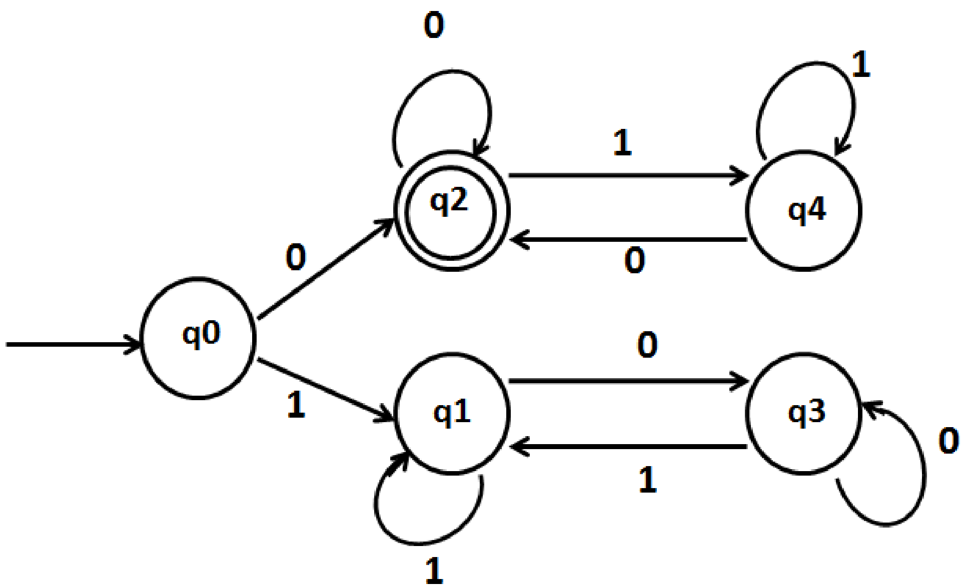
4. نقوم بتحويل NFA الى DFA.



وبعد التحويل من NFA الى DFA حسب الخطوات التي جرى شرحها سابقا نحصل على الآلة التالية:

إذ أن الحالة q0 تقابل المجموعة {0, 1, 4}.
والحالة q_1 تقابل المجموعة {2, 4}.
والحالة q_2 تقابل المجموعة {3, 5}.
والحالة q_3 تقابل المجموعة {2, 5}.
والحالة q_4 تقابل المجموعة {3, 4}.

5. نطبق عملية المتمم مرة اخرى على DFA الناتجة من خطوة (4) لنحصل على آلة DFA تميز تقاطع لغتين منتظمتين إذ نقوم بتحويل كل حالة قبول الى حالة اعتيادية و تحويل كل حالة اعتيادية الى حالة قبول.



11.2 تصميم DFA هي معكوس لآلة DFA اخرى
(Design a DFA that is the reverse of another DFA)

المسألة: اذا اردت ان تصمم آلة DFA لتمييز لغة منتظمة مطبق عليها عامل المعكوس

Reverse Operator او تقرأ كلماتها من اليمين الى اليسار. مثلا، اللغة التي كلماتها w معرفة على الابجدية {0, 1} و من مضاعفات العدد 4 و تقرأ من اليمين الى اليسار وصيغة تعريفها كما يلي:

L= { w | w is a multiple of 4 when interpreted as a binary integer and is read form right to left}.

الحل

1. صمم آلة DFA للغة و هي تقرأ كلماتها بطريقة اعتيادية، اي من اليسار الى اليمين.
2. حول DFA الى NFA تحتوي على الانتقالة Ɛ باتباع الخطوات التالية:
   a. اعكس اتجاه جميع الاسهم التي تمثل الانتقالات بين الحالات.
   b. حول حالة البداية في DFA الى حالة نهاية في الآلة الجديدة NFA.
   c. اضف حالة جديدة مثل p و اربطها بجميع حالات القبول السابقة بالانتقالة Ɛ.
3. تحويل الآلة NFA الى الة DFA.

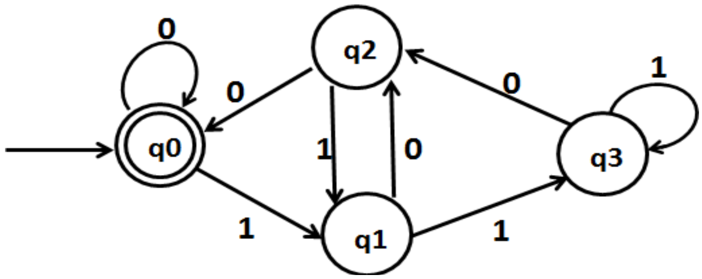
مثال13

صمم آلة DFA تميز اللغة التي كلماتها w معرفة على الابجدية {0, 1} و من مضاعفات العدد 4 و تقرأ من اليمين الى اليسار وصيغة تعريفها كما يلي:

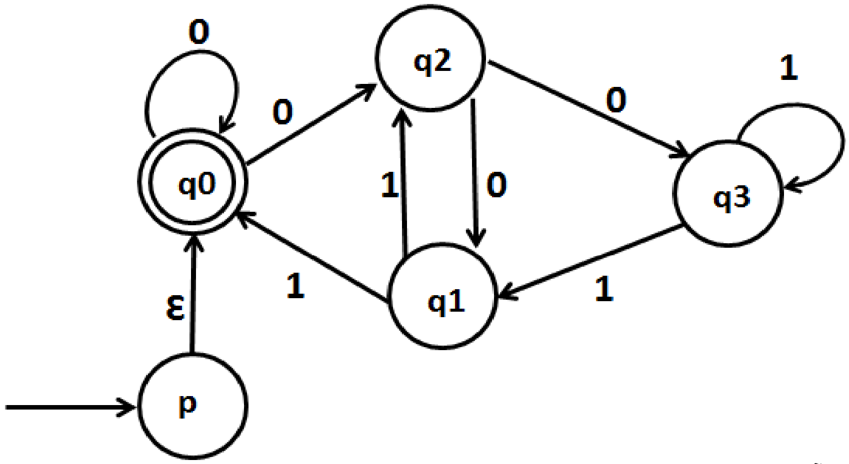
L= { w | w is a multiple of 4 when interpreted as a binary integer and is read form right to left}.

الحل

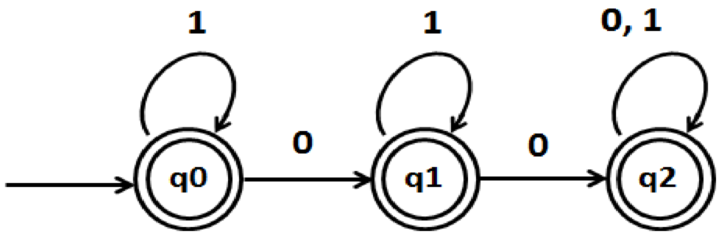
1. صمم الة DFA للغة و هي تقرأ كلماتها بطريقة اعتيادية، اي من اليسار الى اليمين.



2. حول DFA الى NFA تحتوي على الانتقالة ε باتباع الخطوات المذكورة اعلاه.



3. تحويل الآلة NFA الى الة DFA.



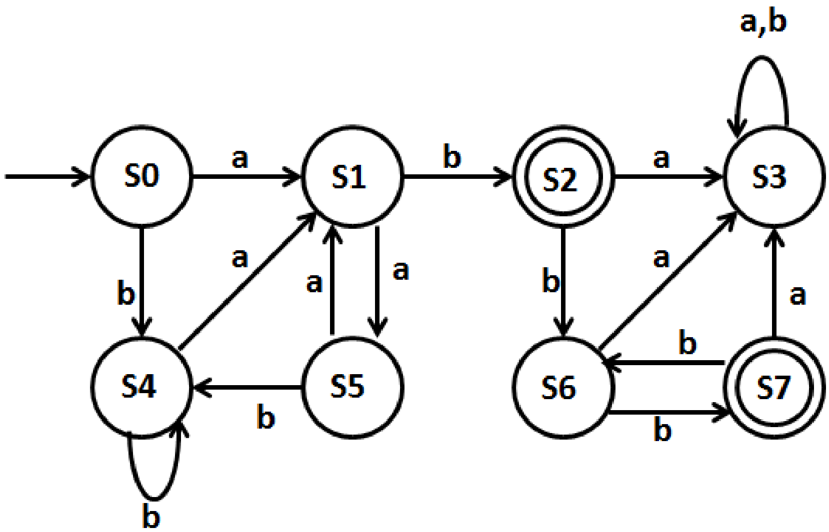
وبعد التحويل من NFA الى DFA حسب الخطوات التي جرى شرحها سابقا نحصل على الآلة التالية:

إذ أن الحالة q_0 تقابل المجموعة {0, 4}.
والحالة q_1 تقابل المجموعة {0, 2}.
والحالة q_2 تقابل المجموعة {0, 1, 2, 3}.

12.2 تصغير حجم الآلة DFA Minimizing DFA

لو نظرنا الى آلة الحالات المنتهية المحددة DFA المعرفة بالشكل ادناه، التي تقبل مجموعة

الكلمات المعرفة بالتعبير المنتظم $(aa + b)^*ab(bb)^*$، نجد انها تتكون من مجموعة من الحالات فيها حالتان نهائيتان(حالتا قبول) هما S2 و S7. وبقية الحالات لا نهائية.



بنظرة فاحصة الى الآلة ، يظهر أن الحالتين S_2 و S_7 في واقع الحال متشابهتان إذ كلتاهما حالتا قبول و كلتاهما يجري الانتقال منهما الى الحالة S_6 عند الادخال b و يكون الانتقال منهما الى الحالة S_3 عند الادخال a. بنفس الطريقة نجد أن الحالة S_0 و الحالة S5 متشابهتان. والسؤال هنا هو لما لا تدمج هذه الحالات و يتولد من هذا العمل آلة اصغر حجما من الاولى (اقل عدد من الحالات) و تؤدي نفس العمل الذي تقوم به الآلة الاصلية.

من هذه الملاحظات، يبدو أن المفتاح لجعل الآلة DFA أصغر هو التعرف على الحالات المتكافئة و دمجها. للقيام بذلك، يجب أن نتفق على تعريف للحالات المتكافئة. الحالات المتكافئة كما عرفها مور Moore هي:

تعريف: حالتان في الآلة M تكون متكافئة اذا و فقط اذا، لاي خيط رمزي x، اذا بدأت الآلة M في واحدة من هاتين الحالتين مع المدخلات x فانها اما تقبل المدخلات في كلتي الحالتين او ترفضها. او بعبارة اخرى ان الآلة M تقوم بنفس العمل عندما تبدأ في اي من الحالتين. و هذا مهم جدا خاصة اذا كانت الآلة تولد مخرجات ، كما في الآلات ذات المخرجات automata with output، مثل آلة مور و آلة ميلي. والسؤال هنا هو كيف نحصل على الحالات المتكافئة؟

حقيقة: الحالات المتكافئة تؤدي الى حالات متكافئة تحت تأثير نفس المدخلات.

نأخذ المثال السابق لتوضيح خوارزمية تصغير الآلةDFA.

1. نقوم اولا بتقسيم حالات الآلة على قسمين: قسم حالات القبول accepting states وقسم حالات الرفض rejecting states. وهذان القسمان هما:

$A = \{s_2, s_7\}, \qquad B = \{s_0, s_1, s_3, s_4, s_5, s_6\}$

إن هذه الحالات متكافئة قبل ان تبدأ الآلة عملها ، اي عندما يكون الادخال يساوي خيطا خاليا empty string.

2. نقوم بالتحقق فيما اذا كانت الحالات في هذه المجموعات تذهب الى نفس المجموعة تحت تأثيرالادخال a وb . كما لاحظنا في بداية حديثنا ان حالات المجموعة A تذهب الى حالات المجموعة B تحت تأثير المدخلات نفسها. بينما الحال يختلف مع حالات المجموعة B. الجدول التالي يوضح نتيجة تأثير المدخلات على هذه الحالات (مثلا، المدخلات a يجعل الآلة تنتقل من الحالة s_1 الى الحالة s_5 في المجموعة B و المدخلات b تجعل الآلة تنتقل الى الحالة s_2 في المجموعة A).

In state	s_0	s_1	s_3	s_4	s_5	s_6
a leads to	B	B	B	B	B	B
b leads to	B	A	B	B	B	A

3. عند النظر الى الجدول نجد ان المدخلات b تساعد في التمييز بين الحالتين (s_1, s_6) و بقية الحالات في المجموعة B لانها تنتقل الى حالات في المجموعة A بدلا من حالات المجموعة B. لذا فان الحالات $\{s_0, s_3, s_4, s_5\}$ لا يمكن ان تكافيء $\{s_1, s_6\}$ وبذلك يجب تقسيم المجموعة B على مجموعتين. تصبح المجموعات كما يلي:

$A = \{s_2, s_7\}, \qquad B = \{s_0, s_3, s_4, s_5\}, \qquad C = \{s_1, s_6\}$

4. نقوم بالتحقق من حالات المجموعة B كما حصل في الخطوة 2 والجدول التالي يوضح نتيجة التحقق.

In state	s_0	s_3	s_4	s_5
a leads to	C	B	C	C
b leads to	B	B	B	B

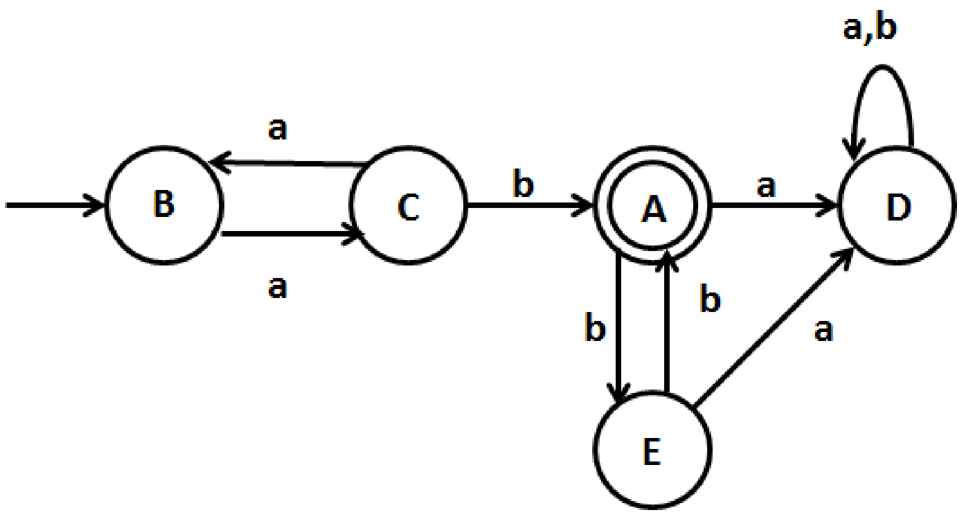
من خلال النظر الى الجدول نجد ان الحالة s_3 لا تكافيء بقية حالات المجموعة وبذلك يجري تقسيم المجموعة B على مجموعتين كالتالي:

$$A = \{s_2, s_7\}, \quad B = \{s_0, s_4, s_5\}, \quad C = \{s_1, s_6\} \quad D = \{s_3\}$$

5. نستمر بتكرار هذه العملية حتى نميز جميع الحالات في مجاميع للتكافؤ. فتصبح المجاميع كما يلي:

$$A = \{s_2, s_7\}, B = \{s_0, s_4, s_5\}, C = \{s_1\}, D = \{s_3\}, E = \{s_6\}$$

و حسب التعريف، فان الحالات في كل مجموعة تكون متكافئة لانها تذهب الى حالات في المجموعة نفسها تحت تأثير المدخلات a و b. والآلة المصغرة تكون كما معرفة ادناه إذ كل حالة فيها هي مجموعة حالات جرى تصنيفها حسب الخوارزمية.



Minimize (M) Algorithm

Input: M

Output: M is the smallest machine

Group A = {accepting states of M}

Group B = {rejecting states of M}

Repeat

For every group

For every state in the group

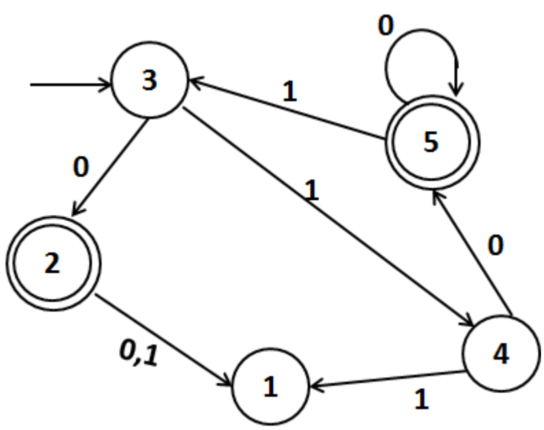
Find which group the input lead to, if there are differences partition the group into sets containing states go to the same group under the input.

Until no partitioning take place

Build a new M with groups as states

اسئلة الفصل الثاني

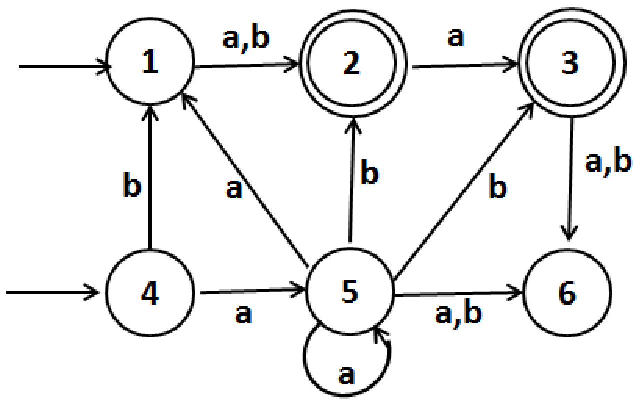
1. ماهي مكونات آلة الحالات المنتهية المحدودة DFA التالية وماهو جدول الانتقالات الخاص بها.



2. ارسم الآلة DFA المعطاة إذ أن:
S={3}, F={2, 4}, K={1, 2, 3, 4}, Σ = {9, R, $} ودالة الانتقال معرفة بجدول الانتقالات التالي:

q	9	R	$
1	{1}	{3}	{2}
2	{4}	{1}	{3}
3	{4}	{3}	{1}
4	{2}	{1}	{2}

3. عرف مكونات الآلة DFA التالية:



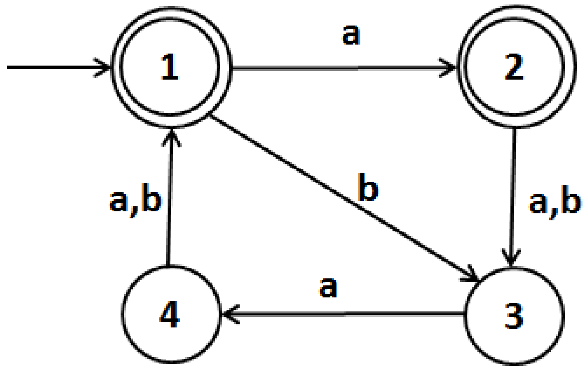
4. ارسم الآلة DFA المعرفة بالتالي:

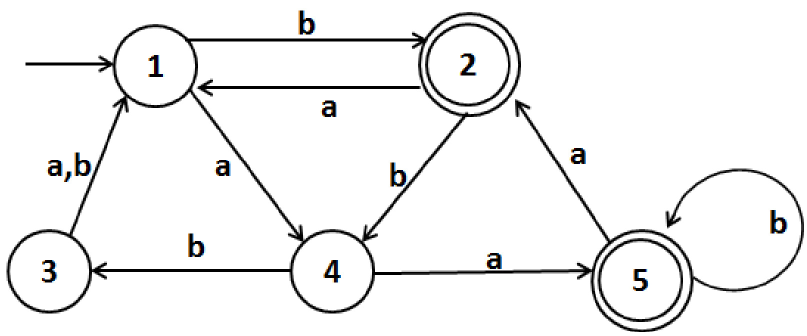
S={3, 4}, F={2, 4}, K={1, 2, 3, 4}, Σ={9, R, +, 5}

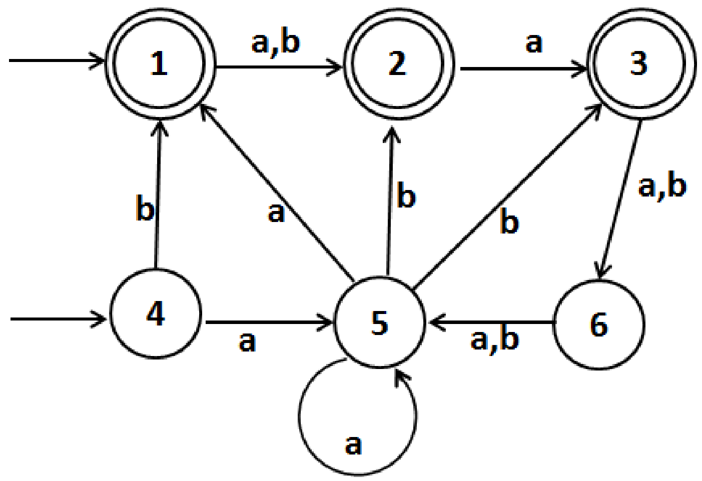
ودالة الانتقال معرفة بالجدول التالي:

q	**9**	**R**	**+**	**5**
1	**{1}**	**{2,3}**	**{2}**	**{1,2,3}**
2	**{1,4}**	**ф**	**{1,3}**	**ф**
3	**{3,4}**	**{3}**	**ф**	**{3}**
4	**{2}**	**ф**	**ф**	**ф**

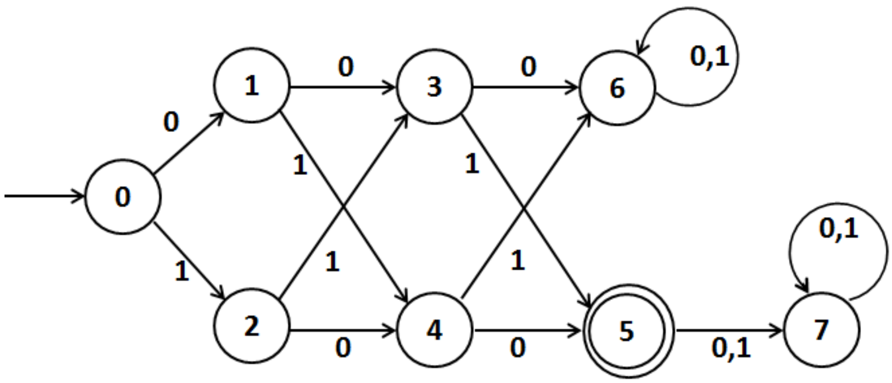
5. حدد فيما اذا الآلات التالية تقبل المدخلات abaa ,ababbaaa ام لا؟







6. عرف اللغة التي تميزها الآلة DFA التالية:



7. آلة DFA معرفة بجدول الانتقالات التالي، إذ أن s0 تمثل حالة البداية و sf تمثل حالة القبول.

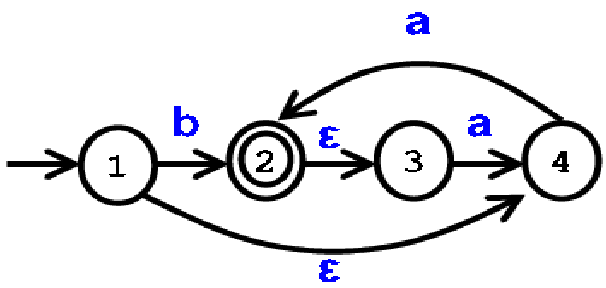
a. ارسم مخطط الحالات للالة.

b. عرف اللغة التي تميزها.

State	0	1
s_0	s_1	s_2
s_1	s_3	s_4
s_2	s_4	s_3
s_3	s_5	s_f
s_4	s_f	s_5
s_5	s_5	s_5
s_f	s_6	s_6
s_6	s_6	s_6

8. عرف الة DFA و ارسمها التي تقبل اللغة L المعرفة على الابجدية {0, 1} بحيث كلماتها تبدأ و تنتهي بالرمز 1.

9. حول الآلة NFA التالية الى الة DFA.



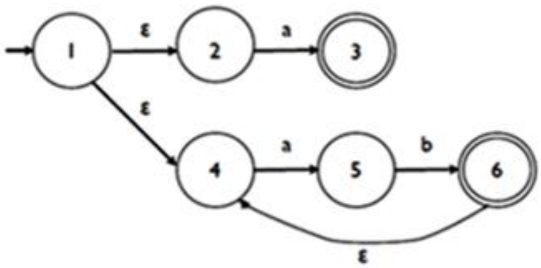
10. صمم آلة DFA تقبل اللغة التالية:

L = {w : w∈{0, 1}* and w when interpreted as a number is not divisible by 3. }

11. صمم الآلة DFA لكل واحدة من اللغات التي خيوطها الرمزية معرفة على الابجدية $\Sigma = \{a, b\}$.

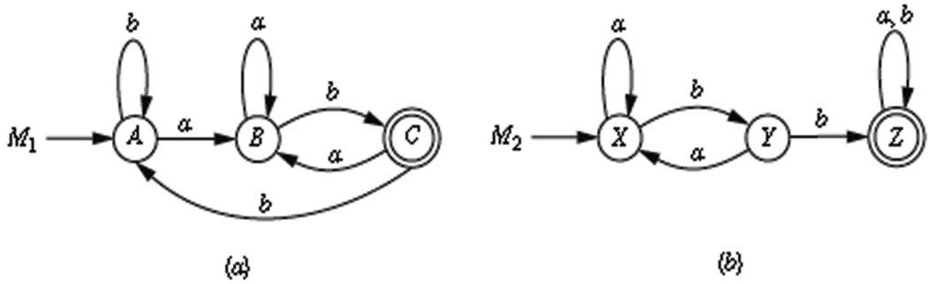
a. $L_1 = \{w \in \Sigma^*: w \neq uaaav,\ u, v \in \Sigma^*\}$
b. L_2 = {w| w has an odd number of a's}
c. L_3 = {w| w has even length}
d. L_4 = {w| w ends with a b}
e. L_5 = {w| w has an odd number of a ' s}
f. L_6 = {w| w has at least two b's}
g. L_7 = {w| w has an even number of a's}
h. L_8 = {w| w has one or two b's}
i. L_9 = {w| w has at most one b}
j. L_{10} = {w| w has at least three a's}

12. اذا كانت الآلة NFA معرفة على الابجدية {a, b}، حول هذه الآلة الى الآلة DFA المكافئة لها.



13. لتكن M_1 و M_2 التين من نوع DFA تقبل اللغات L_1 و L_2 على التوالي. ارسم الآلة DFA التي تقبل كل من اللغات التالية:

a. $L_1 \cup L_2$
b. $L_1 \cap L_2$
c. $L_1 - L_2$



14. صمم الةِ NFA عدد حالاتها ست فقط لتمييز اللغة L المعرفة بالصيغة التالية:

L= {w | w contains an even number of 0s, or contains exactly two 1s}

15. اللغات ادناه تمثل تقاطع لغتين بسيطتين. صمم الة DFA لكل لغة بسيطة ثم ادمجهما لتوليد DFA يميز اللغة. جميع اللغات معرفة على الابجدية {a, b}.

a. {wi w has at least three a's and at least two b's}
b. {wi w has at exactly two a's and at least two b's}
c. {wi w has an even number of a's and one or two b's}
d. {wi w has an even number of a's and each a is followed by at least one b}
e. {wi w starts with an a and has at most one b}
f. {wi w has an odd number of a's and ends with a b}
g. {wI w has even length and an odd number of a's}

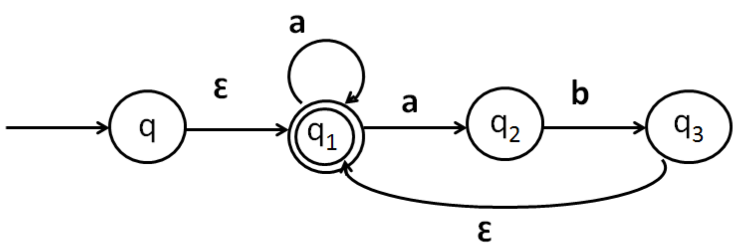
16. جميع اللغات ادناه هي متمم للغة اخرى ابسط. صمم الة DFA للغة البسيطة ثم ولد منها الة DFA لتميز اللغات ادناه.

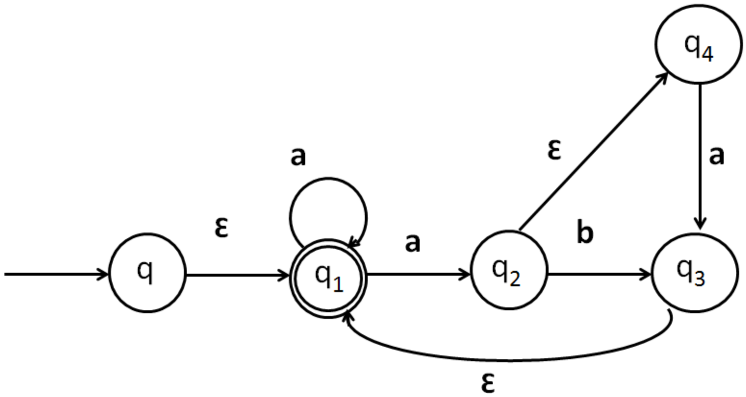
a. {wi w does not contain the substring ab }
b. {wi w does not contain the substring baba}
c. {wi w contains neither the substrings ab nor ba}
d. {wi w is any string not in a * b * }
e. { wi w is any string not in (ab +) * }
f. {wi w is any string not in a* U b*}
g. {wi w is any string that doesn't contain exactly two a's}
h. { wi w is any string except a and b }

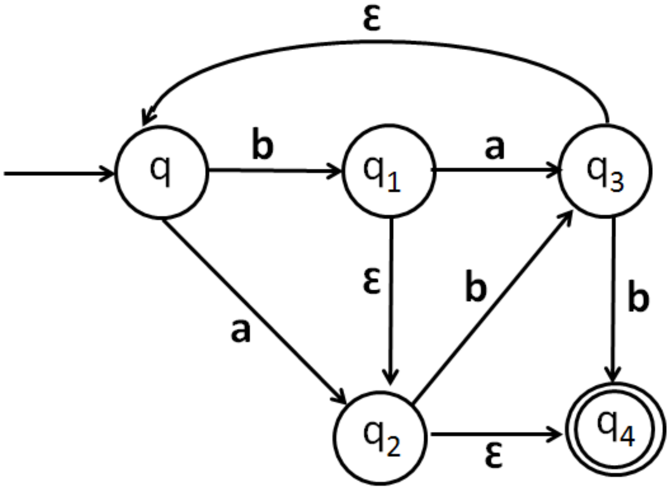
17. صمم الة NFA للغات التالية: (قد تحتوي الآلة على انتقالة من Ɛ)

a. مجموعة الخيوط الرمزية التي تتكون من صفر او اكثر من الرمز a يليه صفر او اكثر من الرمز b يليه صفر او اكثر من الرمز c.
b. مجموعة الخيوط الرمزية التي تتكون من الخيط الرمزي الجزئي 01 يتكرر مرة واحدة او اكثر او من من الخيط الرمزي الجزئي010 يتكرر مرة واحدة او اكثر.
c. مجموعة الخيوط الرمزية التي تنتهي بالرموز aaab.
d. مجموعة الخيوط الرمزية التي يكون فيها من الرمز الرابع الى نهاية الخيط هو الرمز a.

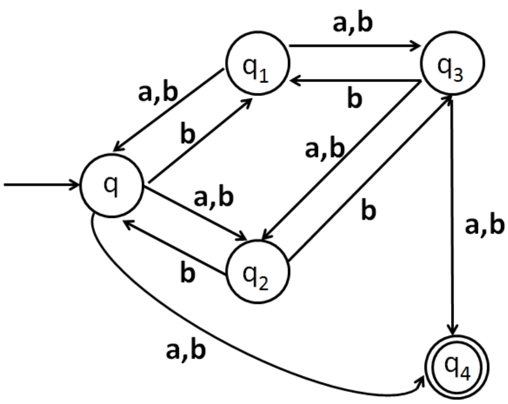
18. حول الآلة NFA التالية الى آلة NFA دون الانتقالة Ɛ.







19. حول الآلة NFA التالية الى الة DFA باقل عدد ممكن من الحالات.



المصادر

1. Maribel FernＬndez, "Models of Computation:An Introduction to Computability Theory", Springer, 2009

2. Anil Maheshwari, Michiel Smid, “Introduction to Theory of Computation”, , 2014

3. S.P.E. Xavier, “Theory of Automata, Formal Languages, and Computation”, New Age International (P) Ltd, 2005

4. Jean-´Eric Pin, “Mathematical Foundations of Automata Theory”, , 2015

5. K.l.P. MISHRA, “Theory of Computer Science:Automata, anguages and Computation”, 3rd ed., Prentice'Hall of India, 2008

6. Eitan Gurari, “An Introduction to the Theory of Computation”, Ohio State University, 1989

7. Willem J.M. Levelt, “An Introduction to the Theory of Formal Languages and Automata”, John Benjamins Publishing Company, 2008

8. J.E. Hopcroft, R. Motwani, “Introduction to Automata Theory, Languages, and Computation”, Addison-Wesley, 2001

9. John C. Martin, “Introduction to Languages and The Theory of Computation”, 4th ed., McGraw-Hill, 2011

10. Andy Dong, “The Language of Design :Theory and Computation”, Springer, 2009

11. Juray Hromkovic, “Theoretical Computer Science:, Springer, 2004

12. Daniel I. A. Cohen, “Introduction to Computer Theory”, John Wiley & Sons, Inc., 1986

13. Peter Linz, “An Introduction to Formal Languages and Automata”, 3rd ed., Jones and Bartlett Publishers, 2001

14. Thomas A. Sudkamp, “Languages and Machines”, 2nd ed., ADDISON-WESLEY, 1997

15. Dexter C. Kozen, “Theory of Computation”, Springer, 2006

16. H.R. Lewis, “Elements_of_the_theory_of_computation”,2nd ed., C.H.Papadimitriou, 1998

17. J.L. Hein, “theory_of_computation._an_introduction “,Jones and Bartlett Publishers, 1996

18. M. Sipser, “Introduction to Theory of Computation”, 2nd ed., Thompson Course Tech., 2006

19. B.M. Moret, “The Theory of Computation”, Addison-Wesley, 1998

20. Gary Haggard, “Discrete Mathematics for Computer Science”, Thomson Brooks/Cole, 2006

21. SUSANNA S. EPP, “Discrete Mathematics With Applications”, 4th ed., Brooks/Cole Cengage Learning, 2011

22. Kenneth H. Rosen, “Discrete Mathematics and Its Applications”, 7th ed., McGraw-Hill, 2012